Universidad Carlos III de Madrid
Escuela Politécnica Superior

Departamento de Informática
Doctorado en Ciencia y Tecnología Informática

Tesis Doctoral

inContexto: A Framework to Obtain People Context Using Wearable Sensors and Social Networks Sites

Gonzalo Blázquez Gil

Dirigida por

Antonio Berlanga de Jesús

José Manuel Molina López

Junio 2015

Address:

Grupo de Inteligencia Artificial Aplicada
Departamento de Informática
Universidad Carlos III de Madrid
Av. de Gregorio Peces-Barba Martínez, 22
Colmenarejo 28270 — Spain

inContexto: A Framework to Obtain People Context Using Wearable Sensors and Social Networks Sites

Autor: Gonzalo Blázquez Gil

Directores: Antonio Berlanga de Jesús

José Manuel Molina López

Firma del Tribunal Calificador:

	Nombre y Apellidos	Firma
Presidente: D.	.....	.....
Vocal: D.	.....	.....
Secretario: D.	.....	.....

Calificación: .....

Colmenarejo, ... de ..... de 2015.

Mamá, Papá y Fer.

Contents

List of Figures

List of Tables

Abstract

AMBIENT Intelligent (AmI) technology is developing fast and will promote a new generation of applications with some characteristics in the area of context awareness, anticipatory behavior, home security, monitoring, Health Care and video surveillance. AmI Environments should be surrounded by multiples sensors in order to discover people needs. These kind of scenarios are characterized by intelligent environments, which are able to recognize inconspicuously the presence of individuals and react to their needs. In such systems, people are conceived as the main actor, always in control, playing multiple roles, and this is perhaps the new real facet of research related to AmI: it introduces a new dimension creating synergies between the user and the environment.

The AmI paradigm sets the principles to design pervasive and transparent infrastructures being capable of observing people without prying into their lives, and also adapting to their needs. There are several basis concepts to consider for retrieving people context, however the most important for users is that sensors devices must be unobtrusive. Many technologies are conceived as hand-held or wearable, taking advantage of the intelligence embedded in the environment.

Mobile technologies and Social Network Sites make it possible to collect people information anywhere at anytime, and provide users with up-to-date information ready for decision-making processes. Nevertheless, the management of these sensors for collecting user context poses several challenges. Besides the limited computational capabilities of mobile devices, mobile systems face specific problems that cannot be solved by traditional knowledge management methodologies and tools, and thus require creative new solutions.

This dissertation proposes a set of techniques, interfaces and algorithms for the implementation of inferring context information from new kind of sensors (Smartphones and Social Networking). The huge potential of both new sensors have motivated us to design a framework that can intelligently capture different sensory data in real-time. Smartphones may obtain and process physical phenomena from embedded sensors (Accelerometer, gyroscope, compass, magnetometer, proximity sensor, light sensor, GPS, etc.) and SNS the affective ones. Subsequently this information could be transmitted to remote locations without any human intervention. The mechanisms proposed here are based on the implementation of a basic framework that

modifies information from the raw data to the most descriptive action. To this end, the development of this thesis starts from a inContexto framework which exploits off-the-shelf sensor-enabled mobile phones and SNS people presence to automatically infer people's context. The main goals of our architecture are: (i) Collection, storage, analyse, and sharing of the user context information, (ii) Plug-and-play support for a wide variety of sensing devices, (iii) Privacy preservation of individuals sharing their data, and (iv) Easy application development. Furthermore our inContexto has been implemented to allow third party application to participate and improve people context.

Abstract

La Inteligencia Ambiental (AmI) está sufriendo una evolución rápida y en un futuro cercano saldrán a la luz una nueva generación de aplicaciones en el área de los sistemas basados en contexto, seguridad en el hogar, monitorización, salud y video vigilancia. Los entornos AmI se caracterizan por estar plagados de sensores los cuales, están encargados de capturar información de la gente que hay en ellos para describir sus necesidades. Este tipo de escenarios se caracterizan por ser entornos inteligentes, capaces de reconocer autónomamente la presencia de personas y reaccionar a sus necesidades. En dichos sistemas, las personas o usuarios se conciben como el actor principal, siempre en control, jugando múltiples roles, y esto es una nueva característica dentro del marco de la investigación relacionada con AmI: introducir nuevas sinergias entre el usuario y el entorno que le rodea. El paradigma AmI establece los principios para el diseño de arquitecturas generales que son capaces de capturar información relevante de las personas sin entrometerse en su vida, y además adaptar dicha información a las necesidades del mismo. Existen diferentes conceptos a tener en cuenta para la captura del contexto de las personas, sin embargo, el factor más importante es que los dispositivos usados deben ser transparentes para el usuario, es decir que trabajen de manera autónoma y sin la ayuda del mismo.

Los nuevos teléfonos móviles inteligentes o *smarpthone* y las redes sociales permiten extraer información de las personas en cualquier lugar en cualquier momento, y así poder proporcionar a los usuarios ayuda para la toma de decisiones en las actividades de su vida real. Sin embargo, la gestión de la información de estos sensores, los cuales nos permiten inferir el contexto, plantean varios desafíos a resolver En primer lugar la limitación de las capacidades tanto computacionales como de disponibilidad (consumo de energía) de los dispositivos móviles, los sistemas móviles se enfrentan a problemas específicos que no pueden ser resueltos por las metodologías y herramientas de gestión del conocimiento tradicional, y por lo tanto requieren de nuevas soluciones creativas.

En esta tesis se propone un conjunto de técnicas, interfaces y algoritmos para inferir la información de contexto de las personas a través de nuevos sensores, los cuales han sido infrautilizados hasta el momento como son los *smartphone* y *Redes Sociales*. Gracias al enorme potencial de estos nuevos sensores nos ha motivado para diseñar un framework que de manera

transparente al usuario puede capturar diferentes datos sensoriales en tiempo real. A través de los Smartphone se puede obtener y procesar los fenómenos físicos (Correr, Andar, etc.) de las personas, utilizando los sensores embebidos como el acelerómetro, giroscopio, brújula, magnetómetro, sensor de proximidad, sensor de luz, GPS, etc. Además a través de las redes sociales se podría obtener información de los fenómenos afectivos del usuario. Posteriormente, esta información se transmitirá para su procesamiento y búsqueda de nuevas inferencias sin la colaboración del usuario, de manera transparente. Los mecanismos propuestos en esta tesis se basan en la aplicación de un framework, inContexto, que recoge la información de los sensores (Señales, palabras, etc.) para posteriormente generar una acción más descriptiva y entendible por el usuario. Los principales objetivos que presenta inContexto son: (i) Recogida, almacenamiento, análisis e intercambio de la información de contexto de usuario, (ii) el apoyo Plug-and-play para una amplia variedad de dispositivos, (iii) la preservación de privacidad de los las personas, y (iv) el desarrollo de nuevas aplicaciones fácilmente, permitiendo a través de inContexto el acceso a los datos a aplicaciones de terceros para mejorar la información recogida.

Agradecimientos

COMO sabéis no soy muy dado a tener muestras de afecto pero aun así lo intentare. Respetando el protocolo empezare por los Popes, Antonio y José Manuel, que confiaron varias veces en mi para realizar el viaje de la tesis. Muchas gracias por todas vuestras críticas y comentarios a mi trabajo. También quería agradecer a todos los profesores de la uc3m que he podido disfrutar durante los 10 magníficos años que pasé allí. En especial agradecérselo a los de mi grupo, el GIAA: Jesús, Javier, MA; Muchas gracias.

En esos 10 años he vivido mucho y con mucha gente, pero los siguientes hieneros han hecho que todo sea mucho más fácil, gracias: Nayat, Luis, José, Rodri, Alberto, Ramon, Kike, Morlis y Mirren.

Aunque amigos tengo muchos, estos son los de toda la vida y gran parte de esta tesis es gracias a vosotros. Irene, Olalla, Javi, Bea, y Ana, Gracias. Esto es lo más parecido a un niño que os voy a poder dar, así que tratadlo como a un sobrino mas. Lo sentáramos en la mesa con Leo, Bicho y los que vengan.

A mis tías/os, primas/os, mis padres, fer, mesa 12, mesa 12 bis, la panda, en general a toda mi familia y especialmente a mis dos abuelas, os quiero un montón.

También muchas gracias a todos los nuevos amigos y compañeros de Coruña que han hecho que mi haya sido mucho más fácil. Y por supuesto a Judith, la culpable que haya cambiado Madrid por un pueblito como Coruña, te quiero.

Gonzalo Blázquez Gil
A Coruña, Junio del 2015.

Introduction

I.1 Antecedents

AMBIENT Intelligence (AmI) aims to enhance the way people interact with their environment to promote safety and to enrich their lives. Ambient Intelligence is represented by electronic environments which are sensitive and responsive to the presence of people. AmI applications and services react specifically to their surroundings, location, and time. The general trend is to integrate seamlessly every electronic device into the environment and change their behavior according to circumstances.

AmI systems are mainly focused on logging environment information however, it neglects how to retrieve context information from people. However, people are the main actor in an AmI world where electronic devices work to support people in carrying out their daily activities, tasks and rituals. Hence, the environment should be able to perceive, recognize and communicate with the individuals who are inside which is called People Context.

AmI line of research has been receiving a special attention in the research community in the last years. While advances in wearable computing have lead to the development of wireless and non-intrusive sensors that can capture the necessary people activity information, current activity recognition approaches have so far experimented on either a scripted or pre-segmented sequence of sensors events related to activities, e.g. ambient assisted living in the case of elders' assistance.

Traditionally, human activity recognition is included into computer vision research field. The main goal of these systems are to analyze automatically ongoing activities from an video source. However, these systems present some drawbacks such as huge computational cost due to complex techniques to interpret video data and also a speedy bandwidth to transmit information in real time is necessary. Besides, in some conditions could be difficult to infer people action, for example, in a crowd place is nearly impossible to determine which concrete action is doing someone. Therefore, recently human activity recognition problem has been faced using wearable sensors and they are becoming extremely popular in the research community. This systems rely on some electronic devices placed across the human body which return a real-time measurement of acceleration along the *x-, y-, or z-axis*. The measurement is used as

a human motion detector and depending on the sensor situation is possible to infer every single person movement.

However, a friendly electronic device is necessary since people are really unwilling to wear one to monitor them. In this point, mobile phones play an important role in the wearable sensors community researcher. Mobile phones are intimately associated with their user's daily life and experiences. The increasing popularity of smartphones with their embedded sensing capability are carrying smartphone sales soar. For example, worldwide sales of smartphones to end users had a record fourth quarter of 2014 with an increase of 29.9 percent from the fourth quarter of 2013 to reach 367.5 million units, according to Gartner, Inc. For the first year ever, in 2011 total PC sales around the world were outpaced by total smartphone sales.

Smartphones are mobile phones that offer not only the standard communication facilities through voice and text, they are also provided by new countless number of sensors (accelerometer, gyroscope, compass, magnetometer, proximity sensor, light sensor, GPS, etc.). The embedded sensors allow to the device adapts to environment conditions, use of battery, lighting conditions, and sound. For example, light sensor controls screen brightness in order to preserve battery life. When the user is using the smartphone in a dark place, the screen brightness is reduced.

Moreover, thanks to this new sensors, it is possible to assert that a smartphone experiences almost the same physical conditions that the person experiences. It feels the same forces, travels at the same velocity, is about the same temperature, is exposed to the same sounds. Hence, if you track mobile phone actions you could track people actions. People are now the carriers of sensing devices however, smartphones opens a new point of view in sensor networks research field.

Smartphones are becoming indispensable in people daily lives thanks to the great amount of amazing apps available on the palm of their hand. In 2007 Apple presents a new large-scale application distribution channel, the first mobile phone application store, where third-party developers can build software and deliver apps for the device. Nowadays, every mobile phone OS has its own Application Store (Nokia Store, Android Market, etc.). A large part of the income earned by the smartphone OS companies come through these online stores.

From that moment, programmers around the world started to create new amazing apps and the most of then use this new embedded sensors which generate countless human data from new sources such as GPS, accelerometers, call logs, Internet connection, etc. This amount of data provides a wealth of new opportunities and it allows us to understand the impact of context on user behavior as well as to study people routines, activity, personality, etc.

Therefore, thanks to smartphones connection over different radio channels it is possible to

consider them as a new sensor inside AmI environments and users finally provides us by sensory abilities. AmI worlds and people are connected through smartphone. Besides, the mobility and power afforded by smartphones allow users to interact continuously with them more than even before. So, they may be considered as a non-intrusive sensor to monitor people lives.

However, not only human activity recognition is relevant in People context applications. User emotional state may be extremely necessary in order to offer good services. Affective Computing or Emotion-oriented computing is a branch of artificial intelligence that deals with the design of systems and devices that can recognize, interpret, and process human affective states (moods and emotions).

Using smartphones ans Social Network Sites as people context sensors is a new proposal that combines some of the approaches mentioned above (AI, AmI, Context awareness, ...) with Soft Computing contributions, Machine learning, in order to offer integral solutions to arise this task. By studying sensing data sets from apps deployed on smartphones and using machine learning techniques in order to analyze the data, we will be able to have better understanding of user and group patterns not possible before. Our proposal, inContexto, provides a common framework, both from a conceptual as well as a practical perspective, for the development of mobile intelligent systems according to people context.

This research is focus on obtaining people context using wearable sensors and social network sites, and also pertains to machine learning and Ambient Intelligence Systems. Trying to minimize the problems inherent in smartphone computation, we studied the architectures and models required to retrieve, manage and store the people context using smartphones, from human physical activities to human emotional state, including group activities and health cares. The remainder of this thesis gives a detailed description about the research and the results obtained and also a rational discussion underlying our proposals, as well as their advantages over previous work.

I.2 Objectives

The general contribution of this dissertation is to create a general framework which collects, infers and store people context based only on data collected using wearable sensors and Social Networks Sites. In this first approach, the proposed framework is focus on provide people context using smartphones. Moreover, a new user context representation is defined in order to cover the new domain that smartphone sensors provide to us. Hence, the proposed architecture, called inContexto, is based on a pyramid which the top level provides more descriptive or symbolic user context information than the low level where people context is represented by

sensors raw data. inContexto aims to provide a common framework, both from a conceptual as well as a practical perspective, for the development of mobile intelligent systems taking into account people context from different sources. Accordingly, the specific objectives of this work can be summarized as follows:

1. To review and analyze the state of the art in obtaining people context using wearable sensors and related research areas.
    - To examine the problems inherent in people context and related research areas.
    - To analyze the benefits provided by people context systems in different application domains.
    - To review relevant contributions from areas such as Mobile Computing, Ambient Intelligence and Ubiquitous Computing.
2. To propose an abstract architecture to support retrieving raw data from the sensors in order to recognize people context.
3. To define a set of basic physical actions and emotional states of a common user and also to propose a representation there of, modular, scalable and simple. Besides, this thesis aims to has so far focused on recognizing simple human activities. Recognizing complex activities remains a challenging and active area of research. Human activities poses the following challenges:
    - Recognizing complex activities. People can do more than one activities at the same time, such as watching television while talking to friends. These behaviors should be recognized using a different approach from that for sequential activity.
    - Recognizing interleaved activities. Certain real-life activities can be interleaved. For instance, if a friend calls while you are cooking, you'd talk to your friend for a while, while you continue to cook.
    - Ambiguity of interpretation. Similar situations can be interpreted differently. For example, an open refrigerator can belong to several activities, such as cooking or cleaning.
    - Multiple residents. More than one resident can be present in many environments. A smart space, for example, a smart house needs to recognize the activities residents perform in parallel, even when a group performs them.
4. From the data obtained through the smartphone and sensor devices create a classifier that will be able to establish the context of the current user.

5. Create an API infrastructure to store and share the states of each of the people who have the application. Within this API, you can create services to third parties with basic information user context. This architecture must be modular, scalable, etc. to include new data or create new data.

6. Finally, a study case is proposed in order to validate the proposal framework of this thesis. The study case was perform in an eHealth scenario.

I.3 Thesis Structure

Throughout the description of our research, we clearly distinguish between original contributions, i.e. new proposals formulated in this thesis, and the contributions of others in related disciplines. Chapter 2 presents the main contribution of this thesis, namely, inContexto, an abstract architecture for inferring people context from multiple sources. Besides, the evolution of this context information is a key point in our proposal where it evolves from sensor raw data to descriptive context information. The rest of this document is structured as follows:

Chapter 1 offers a panorama of Context-Aware Systems by describing most common definitions and architectures. Also, it gives an overview of related work in the area of activity recognition with wearable sensors. We give a brief historical perspective and then review different applications, sensors, activities and machine learning approaches that have been proposed.

Chapter 2 introduces a novel architecture for collecting people context. It suggests a strategy for evolving data from sensors to high level activities. The abstract architecture is composed by for levels where user context information is evolving to a more descriptive user information. It introduces a novel approach for unsupervised learning of activities from low-level sensor data.

Then it goes with a strategy for selecting those activities in order to select the most descriptive user information. This is achieved by filtering user actions by social, time and position features. Finally, Chapter 2 takes a step towards recognition of high-level activities from Social Networks and intelligent mobile phones.

Chapter 3 presents the design and implementation of level 1 inContexto approach to infer user emotional and psychical activity. An evaluation carried out among users using the application developed to look into the impact of proactive context frameworks and the usability of these is detailed. As a result, significant outcomes regarding the factors that have to be considered when designing a people contexts framework in terms of user efficiency are explained.

Chapter 4 general results from the project involved is presented along with the validation of every contribution. The results of this dissertation from the point of view of scientific results as dissemination into the research community are also presented, apart from additional information on the activities associated with this dissertation.

Chapter 4.5 summarizes the work of the thesis, draws conclusions of this PhD dissertation. Besides it gives an outlook to possible future work with their origin on what has been presented here.

A number of complementary appendixes are included mainly for reference purposes. In particular, Appendix A lists the publications related to this work. Appendix B describes a proposal of a emotion dataset using mobiles phones and social network sites.

1

State of the art

In this chapter, a general background of the design challenges related to mobile computing is provided. First of all, the relationship with other technologies like Context Awareness, Ambient Intelligence systems and Multimodal architectures are stated. Afterwards, activity and emotional recognition systems are presented as a first approach to retrieve inconspicuously the information context of the user with wearable sensors. Finally, the multimodal architecture interaction paradigm is put forward as an intuitive way of accessing the information.

1.1 Context-Aware Systems

Nowadays, users are changing the way they interact with computer devices. Mobile devices that allow users to connect anytime and anywhere instead of a fixed place are gradually superseding personal computer. Besides, the great expansion of wireless networks and also smart mobile phones allow context aware researchers to satisfy easily user's needs. Wireless networks and smart mobile phones technologies allow users to move freely with computing power and network resources at hand and also present new paradigm called Mobile Computing.

Mobile computing field is closely linked to the pervasive computing research field (Weiser, 1991) and context aware systems. On the one hand, pervasive computing attempts to integrate electronic devices (sensors, computing devices and so on) into the environment, enabling users to obtain services every time and everywhere, while, on the other hand, context aware systems aim to obtain information about the user's circumstances (location, time and activity) and surroundings. The concept of context aware systems is among the one of the most exciting trends in computing today.

In general, context aware is represented by applications which change their behavior according to the conditions around them; in this case the smartphone conditions. Applications

and services react specifically to their surroundings, location and time. Particularly, using mobile devices, user context data changes rapidly, because of the user are moving and the location and theirs surroundings are every time different. Therefore, it is desirable that services can react specifically according to the changing user circumstances and adapt to user behavior. Hence, the computing paradigm in which applications can discover and take advantage of contextual information is the so-called context aware systems.

The history of context-aware systems started in 1994 when the term *context-aware* was introduced by Schilt, Theimer (Schilit and Theimer, 1994). However, in 1992 Want et al. (Want et al., 1992) created Active Badge Location System which than were considered to be the first context-aware applications. From that moment, many context aware systems have been created, for example, in late nineties two location aware systems were developed (Sumi et al., 1998) and (Abowd et al., 1997). These systems use location information in order to provide tourist information near them. Another important milestone in context aware system is (Muñoz et al., 2003) which presents one of the first context aware systems in a health-care environments.

Next subsections will discuss some relevant points in context-aware systems. First of all, we give the most relevant definitions of context considering the community research and we chose which one is the most suitable for our system. After we state clear the most referred context-aware architectures and the special requirements to provide these context-aware services in AmI scenarios.

1.1.1 Context Definition

In the literature there are a lot of researchers who try to formally define the meaning of context. A large number of definitions of context and context-awareness has been proposed in the area of computer science, although most of those are not entirely precise in the mobile computing scenario. Merriam-Webster's Collegiate Dictionary definition of context is *"the interrelated conditions in which something exists or occurs"*. While this is a general definition, it does not help much for understanding the concept in a computing environment. On the contrary, the first formal definition of context was given in 1994 by Schilit and Theimer (Schilit and Theimer, 1994). They defined context as:

"a software which adapts according to itself location of user, the collection of nearby people and objects, as well as changes to those objects over time".

The description given by Schilit and Theimer is also very general, however they posed the main foundations of context aware systems in their definition. Defining context-aware systems

as a collection of mobile and stationary computing devices which are able to communicate and cooperate on behalf of a user. Moreover, the authors describe four aspects which every context-aware applications should have:

1. Proximate selection, a user-interface technique where the objects located nearby are emphasized or otherwise made easier to choose.
2. Automatic contextual reconfiguration, a process of adding new components, removing existing components, or altering the connections between components due to context changes.
3. Contextual information and commands which can produce different results according to the context in which they are issued.
4. Context-triggered actions, simple IF-THEN rules used to specify how context-aware systems should adapt.

Subsequently, some other researchers try to formally define context, for example, Schmidt et al. (Schmidt et al., 1999a) define context as knowledge about the user's and information technologies device's state, including surroundings, situation and location. While location information is by far the most frequently used attribute in context definition is not the unique way to characterize people context. Attempts to define context information have grown over the last few years and Dey (Dey and Abowd, 2000) in 2000 gave one of the most accurate context definition:

Any information that can be used to characterize the situation of entities (i.e., whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves.

According to Dey definition everything in the world may be considered as an entity, for example, a bedroom, chair or even a table has its own context. Here, for example, some entities are describing according Dey thesis dissertation.

- Places are regions of geographical space such as rooms, offices, buildings, or streets.
- People can be either an individual or a groups of people.
- Objects are either physical objects or software components and artifacts, for instance an application or a file.

Moreover, Dey in his thesis dissertation also explains that there are some types of context more important than others. Location, identity, time, and status are the most important context types for characterizing the situation of a particular entity and it was described as follows:

1. Identity refers to the ability to assign a unique identifier to entity. The identifier has to be unique in the context aware application.

2. Location is more than just a concrete point in the space. It is expanded to include orientation and elevation, as well as all soft location information which is possible to deduce spatial relationships between entities, such as proximity, or containment, etc.

3. Status characterize the entity that can be sensed. However, these characteristics are different depending on the type of entity and the context aware application. For a place, temperature, the ambient light, noise level may be considered as a status. For a person or group, it can refer to physiological factors, or the activity the person. For software components, status basically refers to any attribute of the software component (Files, CPU workload).

4. Time information helps to characterize a situation which allow us to uses entity historical information. Normally, it is necessary to use in conjunction with other kind of context, either as a timestamp or indicating an instant or period during which some other contextual information is relevant. Although, in some other cases, just knowing the relative ordering of events are enough.

Nevertheless, Day context is currently obsolete due to the evolution of this line of research. For that reason in 2007, Zimmermann et al. (Zimmermann et al., 2007) presents another and more accurate definition of context. They believe that the current definitions of context fail to establish any fundamental basis for their construction, since they are basically driven by the ease of implementation. Hence, they completed the most accurate definition (Dey context definition) introducing two extensions that provide a natural understanding of this concept to users of context-aware applications. This extensions are related to formal and operational definition of context On the one hand, the operational definition characterizes the use of context and its dynamic behavior and on the other hand a formal definition which describes the appearance of context. Hence, the new context definition says:

Context is any information that can be used to characterize the situation of an entity. Elements for the description of this context information fall into five categories: individuality, activity, location, time, and relations. The activity predominantly determines the relevancy of

context elements in specific situations, and the location and time primarily drive the creation of relations between entities and enable the exchange of context information among entities.

The resulting definition of context puts each entity in the centre of a surrounding individual context as we can see in Figure 1.1 which shows a graphical representation of the new contribution of context definition provided by Zimmermann et al.

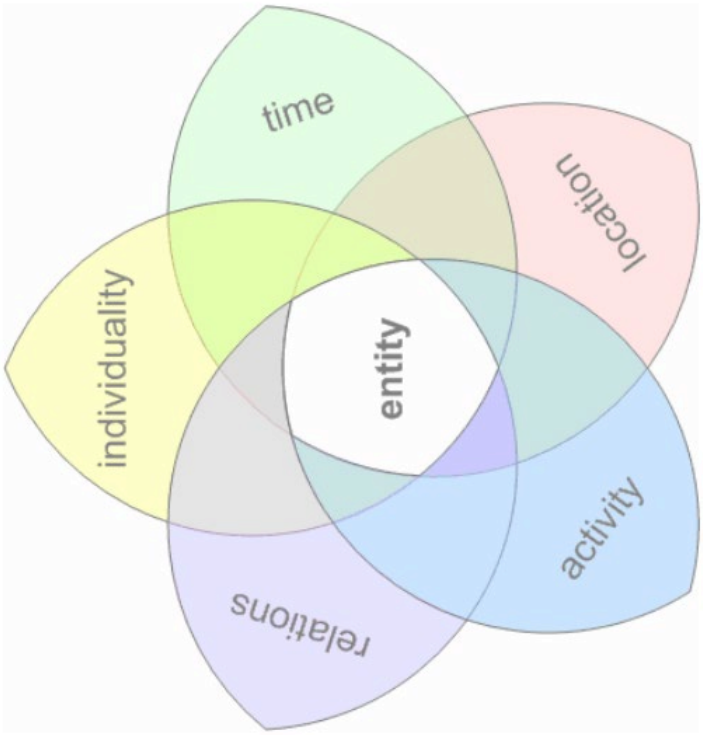

Figure 1.1: Zimmermann et al. representation of an Entity including activity, time, location, identity or individuality and social realtions..

1.1.2 Context Architectures

There are many ways to implement a Context-aware depending basically on acquisition of context-data information. Sensor location (Local or remote), the amount of users, the kind of devices (Smartphones, PC or small mobile devices) are some points to take into account when we design a Context aware system. Chen (Chen, 2004) presents three different approaches on how to acquire contextual information.

- Direct sensor access. Thanks to the rapid advancement of size, wireless connection and battery life of sensors can be directly access by pervasive context-aware systems. The client software only gathers the desired information directly from these sensors. However, sensors needs to be physically connected into the application, so it is hardly difficult to find this kind of context aware systems, although nowadays thanks to the embedded sensors into smartphones are becoming more and more popular. A key benefit of this approach allows to the high-level applications control over the low-level sensors.

  Otherwise, direct sensor access has some shortcomings in distributed systems since the application must maintain the communication with every single sensor. Besides, as much complete is the context, the amount of sensors and the communication process also increases. The most relevant works in direct sensor access in context aware architectures field are (Hinckley et al., 2000) and (Want et al., 1992).

- Middleware infrastructure. Middleware architecture was created mainly to overcome the shortcomings of direct sensor access. The middleware-based approach introduces a layered architecture to context-aware systems with the intention of hiding low-level sensing details to the developers. The idea is that using middleware infrastructures, applications can focus on how to use context instead of how to acquire it.

  Extensibility and reusability are the strong points of Middleware infrastructure compared to direct sensor access since communication between middleware and context application is pre-defined. However, middleware context acquisition approach imposes additional computational costs (CPU power, memory, network bandwidths, etc.). Odyssey project (Noble et al., 1997) and Context toolkit (Dey, 2000) are the most known developments using Middleware infrastructure.

- Context server. This distributed approach takes a step forward by adding multiple and simultaneous connection features in order to manage sensor context information from remote data sources. Gathering sensor data now is placed into the context server to facilitate concurrent multiple access which provides contextual information to different

context aware applications in a distributed environment. This solution reduces the impact in mobile devices which are probably the majority of devices used in context aware systems with huge handicaps in battery life, computation power, disk space, etc. The Me-Centric Domain Server developed by Perich (Perich, 2002) is a good example of a context server architecture.

1.1.3 Context Modeling

A well-designed model is normally a key accessory to the context in any context-aware system. So far, however, there has been a great challenge to describe contextual facts and interrelationships in a precise and traceable way. It is becoming increasingly difficult to ignore the importance of a well-designed model to describe people context. A large number of context-aware applications based on various context models have been developed over the years for a variety of application domains. A good context information modeling formalism reduces the complexity of context-aware applications and improves their maintainability and availability.

In addition, since gathering, evaluating and maintaining context information is extremely expensive, re-use and sharing of context information between context-aware applications should be considered from the beginning. In the next lines the most representative context modeling techniques are described.

1.1.3.1 Key-Value Models

The model of key-value pairs is the most simple data structure for modeling contextual information. Key-value models define the list of attributes and their values which describe context information used by a context-aware application. The key-value modeling approach is frequently used in distributed frameworks where the services itself are usually described with a list of simple attributes in a key-value manner, and the employed service discovery procedure operates an exact matching algorithm on these attributes.

The W3C standard for description of mobile devices, Composite Capabilities/Preference Profile (CC/PP) is probably the first context modeling approach to use Resource Description Framework (RDF) and to include elementary constraints and relationships between context types. In particular, key-value pairs are easy to manage, but lack capabilities for (i) capturing different types of context, (ii) relationships and dependencies between entities, (iii) timeliness and (iv) supporting reasoning on context, on context uncertainty and on higher context abstractions.

1.1.3.2 Object-role models

The object-role based modeling (ORM) approach was originated from information systems modeling to provide an easy mapping from real world context concepts to modeling constructs. The approach also uses a novel form of predicate logic to reason about high-level context abstractions and aims, in particular, to satisfy the heterogeneity, timeliness, reasoning and usability requirements.

Moreover, the intention to employ the main benefits of any object oriented approach - namely encapsulation and reusability - to cover parts of the problems arising from the dynamics of the context in ubiquitous environments. The details of context processing are encapsulated on an object level and therefore hidden to other components. Specific interfaces are provided in order to access to contextual information.

Although there are plenty ORM, this section is focused on context modeling approaches which rely on database modeling techniques. In particular, it concerned with the Context Modeling Language (CML) which was described in a preliminary form in (Henricksen et al., 2002) by Henricksen et al. in 2002. CML provides a graphical notation designed to support the software engineer in analyzing and formally specifying the context requirements of a context-aware application. The formality of ORM and the CML extensions makes it possible to support a straightforward mapping from a CML-based context model to a runtime context management system that can be populated with context facts and queried by context-aware applications.

1.1.3.3 Spatial models of context information

As it is previously said space was considered the most important context in many context-aware applications. For example, Dey et al. (Dey and Abowd, 2000) definition about context, space can be seen as a central aspect of context entities. Thus, some context modeling approaches give space and location a preferential treatment. Location information is obtained by positioning systems which track mobile objects and report their position to a location management system. Mainly, two types of coordinate systems are supported:

- *Geometric coordinates*: Represent a points or areas in a metric space. The most common coordinate system to use is the WGS84coordinates of GPS(latitude, longitude, and elevation above sea level), where each point has an latitude and longitude representing its horizontal position, and a elevation above sea level representing its vertical position.
- Symbolic coordinates: As also known as Cell-ID are represented by an identifier of the place. In contrast to geometric coordinates there is no spatial relation offered by symbolic

coordinates. In order to allow spatial reasoning about inclusion (for ranges) and distances (for nearest neighbors) explicit information about the spatial relations between pairs of symbolic coordinates has to be provided.

Finally, context may be considered as a specific kind of knowledge. Thus, it is quite natural to investigate if any known framework for knowledge representation and reasoning may be appropriate for handling context.

1.2 Recognition of Individual Activities

There exist a wide range of activities that have been monitored and recognized in related work. Nevertheless, a definite and commonly used categorization of them is not provided yet. Huynh (Huynh, 2008) presents a possible way to categorize activities by grouping them based on duration and complexity. This categorization mainly defines 3 different groups:

- *Gestures*: brief and distinct body movements. Gesture recognition is a large research topic in itself, but is not within the focus of this thesis.
- *Low-level activities*: sequence of movements or a distinct posture.
- *High-level activities*: A collection of low-level activities.

Gesture recognition although it is a large research topic in itself, it is not within the focus of this thesis. Thus, we will only focus on low-level and high-level activities. A brief overview of related work performed for low-level and high-level activity monitoring will be given in following subsections.

1.2.1 Low-Level Activities

The monitoring and recognition of low-level activities is common researcher topic. It has been shown as reliable the recognition of a few activities (usually transportation modes, locomotion activities and postures) is possible with few sensors (3D-accelerometer, video cameras, etc.). An example is given by Lee et al. (Lee et al., 2003), where only a tri-axial accelerometer is used to create a real-time personal life log system, based on activity classification. The authors selected 7 activities to be distinguished: lying, sitting, standing, walking, going upstairs, going downstairs and driving. Further examples of recognizing a few low-level activities with just one

3D-accelerometer are given in (Ermes et al., 2008; He et al., 2008; Long et al., 2009; Tapia et al., 2007).

Nowadays, a popular research topic is the monitoring of low-level activities while only using sensors provided by smartphones. For example, Kwapisz et al. (Kwapisz et al., 2011) recognize the activities walking, jogging, going upstairs, going downstairs, sitting and standing by using the embedded accelerometer into a cell phones. However, using smartphones for user activity monitoring produces several new challenges, for example, the mobile phone situation whether it is in the user's hand, in the user's pocket or in a bag carried by the user. Besides, the mobile phone orientation is not predetermined during a regular usage which has to be taken into account in the training phase of such applications. Another important point to take into account is the fact that activity-monitoring applications constantly drain the battery of mobile phones which could limit the regular use of these devices.

Therefore, energy-efficient solutions have recently been investigated for activity recognition on mobile phones, presented e.g. in (Gordon et al., 2012). A further topic of interest nowadays is the recognition of a wide range of low-level activities with multiple sensors, placed on multiple locations of the user's body. For example, Patel et al. (Patel et al., 2009) use 10 Shimmer nodes (each including an accelerometer and a gyroscope) to distinguish different activities performed during gym exercising and the user's daily routine. Another example is presented in (Alimoglu and Alpaydin, 1996): 19 activities (mainly aerobic sport activities, such as cycling, rowing or exercising on a cross trainer) are recognized using 5 inertial measurement units. Since this thesis mainly focuses on the monitoring of low-level activities, mostly on the recognition of a wide range of activities with multiple sensors, further examples of related work on

1.2.2 High-Level and Long-Term Activities

The majority of work in activity recognition focuses on short-term activities that can be measured in minutes or seconds (Walking, jumping, running, etc.), comparatively than hours or even days. As we previously described, the majority of research in activity recognition focuses on rather low-level and short-term activities. However, in many applications ranging from military to assisted living, the analysis and recognition of high-level and longer-term activities is an important component.

In the following sections, we discuss related work in the area of activity recognition and discovery, with a focus on authors aiming towards high-level activities. An example on how the recognition of low-level activities can be utilized for detecting high-level activities is shown by Huynh et al. (Huynh et al., 2007). They complete a dataset where 3 high-level activities;

preparing for work, going shopping and doing housework; are included. Each of these activities are composed by a set of low-level activities (driving car, walking, working at computer, waiting in line in a shop and strolling through a shop, etc.).

As low-level activities researchers consider activities such as walking, sitting, standing, vacuuming, eating, washing dishes, etc. In general, activities which can be characterized by a sequence of body motion, posture or object, use, and which typically, last between seconds and several minutes. High-level activities, on the contrary, are usually composed of a collection of low-level activities, and are longer-term as e.g. cleaning the house, travelling which will typically last more than several minutes and can last as long as a few hours.

On larger time scales, the types and properties of activities are different from those on smaller scales which poses challenges to existing recognition approaches. For instance, activities such as working or going shopping gather a few short-term activities which are performed in changing order and can possibly overlap. They also exhibit a larger variance in execution than short activities. Location and time of day become more important but are often not enough to reliably characterize an activity: there is a large variability in human activities, and different activities can be performed at the same location (e.g. holding a meeting in a restaurant, having lunch at the office desk, etc.).

High-level and longer-term activity recognition has great possibilities in multiple areas such as medical diagnosis and human behavior modeling for instance. However, detecting long-term activities presents several challenges to take into account. Long-term recordings require new annotation techniques that minimize the burden on the user while still attaining sufficiently detailed ground truth. Moreover, they cannot be deployed in laboratory settings but must be conducted "in the wild", i.e. in everyday environments which requires robust and power-efficient hardware. Finally, they require efficient algorithms in order to deal with possibly large amounts of data.

Below, some of the most relevant works in High-Level and Long-Term activities are described. Clarkson et al. (Clarkson and Pentland, 1999) present an approach for unsupervised decomposition of data from on-body sensors into events and scenes. They use data from wearable sensors to discover short events such as "passing through a door" or "walking down an aisle", and cluster these into high-level scenes such as "visiting the supermarket" by using hierarchies of HMMs. Conceptually, this approach relies on high-dimensional and densely sampled audio and video streams. A major criticism of Clarkson et al. work is that cameras and microphones are used and normally they are often considered intrusive, such an approach will be difficult to put into practice.

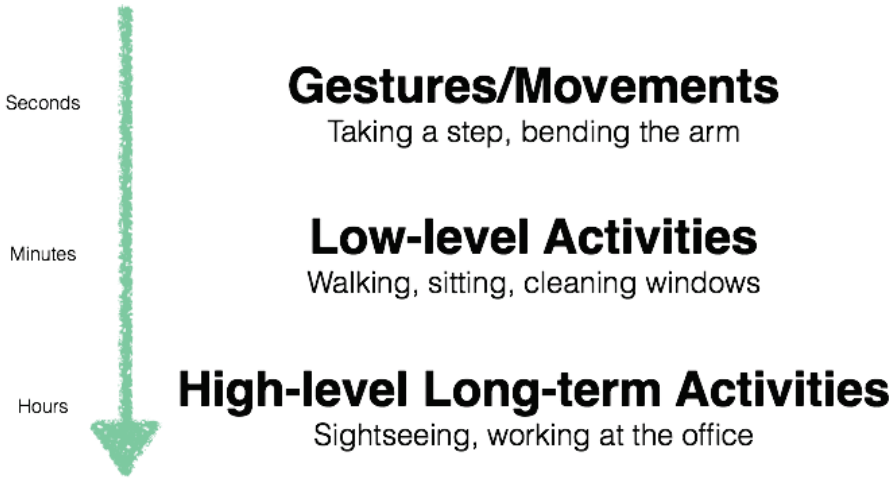

Figure 1.2: Activity categorization based on duration and/or complexity.

Another work, Eagle et al. (Eagle and Pentland, 2006) used coarse-grained location and proximity information from mobile phones to detect daily and weekly patterns of location transitions. Besides, their work is focused on group activities rather than the individual and explores themes such as social networks and organizational rhythms.

In Human Resources application Oliver et al. (Oliver et al., 2002) implements a layered HMM representation to infer office activities such as giving a presentation, having a conversation or making a phone call, based on low-level information from audio and visual sensors as well as from the user's keyboard and mouse activity. In a similar setting, (Horvitz et al., 2002) combine device usage with calendar data and time of day/ time of week information to infer a user's availability. Begole et al. (Begole et al., 2003) analyze and visualize daily rhythms of office workers by measuring how active (indicated by computer usage) a person is during different times of day.

Finally, we aim to highlight those works that rely on location sensors since there is a significant amount of them. Researchers in this field often use terms such as high-level activity when referring to more meaningful descriptions of low-level position information (such as latitude

and longitude), the latter being difficult to interpret by humans. This is slightly different from the point of view taken in this thesis, in which we consider high-level activities rather as a collection of related low-level activities.

An example of work that uses location sensors is given by Liao et al. (Liao, 2006), who use information from GPS sensors to construct models of high-level activity (such as work, leisure, visit) and to identify significant places. Similarly, Krumm and Horvitz (Krumm and Horvitz, 2006) use location sensors to make high-level predictions about driving destinations. These works show that location is a powerful cue to the high-level structure of daily life. However, location is often not enough to identify daily routines reliably, as different activities can be performed at the same location. E.g. at home, many people are having dinner and breakfast but also perform work. Similarly, in an office room one might work, hold meetings and even occasionally have lunch. Therefore, we consider the work that we describe in Chapter 7 complementary to these approaches, in that the use of accelerometers allows detection of more fine-grained activities and can also account for different activities performed at the same location.

Furthermore, with the knowledge obtained about an individual's high-level activities, the recognition of his low-level activities can be improved. An example on how the recognition of low-level activities can be utilized for high-level activity recognition is shown by Huynh et al. (Huynh et al., 2007). They recorded a realistic dataset including 3 high-level activities: preparing for work, going shopping and doing housework. Those activities are composed of a set of low-level activities, for example going shopping consists of driving car, walking, working at computer, waiting in line in a shop and strolling through a shop. Both low-level and high-level activity labels are given in their recorded dataset. The authors used simple features and common algorithms (kNN, HMM and SVM). One of their key findings was that the recognition of high-level activities could be achieved with the same algorithms as the recognition of low-level activities. Moreover, they could distinguish between the 3 defined high-level activities with a recognition rate of up to 92%.

The work presented in (Huynh et al., 2008) uses topic models to recognize daily routines as a probabilistic combination of activity patterns. First of all, the authors recognize a large set of low-level activities using a wearable sensor platform. By using this information, they show that the modeling and recognition of daily routines is possible without user annotation. Overall, 4 daily routines (high-level activities) were recognized: commuting, office work, lunch routine and dinner activities.

1.2.3 Activities of Daily Living

A specific set of activities, called activities of daily living, was first proposed by Katz et al. (Katz et al., 1963) in order to provide a standardized way to estimate the physical well-being of elderly and their need for assisted living. The following activities are included in the set of ADLs: bathing, dressing, toileting, transferring, continence and feeding. Moreover, Lawton and Brody (Self-maintenance, 1969) proposed another set of activities, called instrumental activities of daily living (IADL), in order to assess how well elderly interact with the physical and social environment. The set of IADLs consists of the following activities: using telephone, shopping, food preparation, housekeeping, doing laundry, transportation, taking medications and handling finances.

Various approaches exist for the monitoring and recognition of specific subsets of ADLs/IADLs. For example, Stikic et al. (Stikic et al., 2008) combine RFID tags and accelerometers to recognize 10 housekeeping activities (such as dusting, ironing or vacuum cleaning). With these sensing modalities they combine two main assumptions related to ADLs: 1) the objects people use during the execution of an activity robustly categorize that activity (RFID tags) and 2) the activity is defined by body movements during its execution (accelerometers). The authors of (Huynh, 2008) use a wrist-worn device to distinguish 15 ADLs. This device includes the following sensors: accelerometer, microphone, camera, illuminometer and digital compass. Results show that the camera is the most important single sensor in the recognition of this subset of ADLs, followed by the accelerometer and the microphone.

The work by Maekawa et al. (Maekawa et al., 2012) introduces the concept of mimic sensors: the mimic sensor node has the shape of objects like a AA battery or an SD memory card, and provides the functions of the original object. Moreover, these sensor nodes provide additional information (e.g. current flow of the device) which can be used to detect electrical events. These events can then be used to recognize ADLs such as shaving or vacuum cleaning. Further recent examples of research work on the topic of monitoring and recognizing ADLs/IADLs are presented e.g. in Reiss et al. (Reiss, 2014).

1.2.4 Activity Recognition Using Accelerometers

The ability of human activity recognition seems so natural and simple for us, however; actually it requires complicated functions of sensing, learning inference and classification for computers devices. There are mainly two ways of deal with human activity recognition problem; the first one is computer vision and second one using wearable sensors. Wearable sensors have started becoming useful in a wide variety of environments, and there have been numerous research

efforts that show how these sensor networks can be applied to such areas ranging from everyday situations to scientific pursuits.

Related work in activity recognition using wearable can be grouped based on the types of systems used to implement the algorithms: commercial devices, custom hardware, and mobile phones (Consolvo et al., 2008). Every activity recognition architectures are briefly described below.

1.2.4.1 Commercial Devices and Custom Hardware

Research in activity recognition using inertial sensors commenced long before smartphones hit the market. Initially, tracking user movements wearing ad-hoc accelerometers across the body (Custom hardware) and more recently using pedometers in commercial devices. These platforms extended the idea of wearable computing with senses that served to bridge the gap between the physical and digital.

On one hand, Commercial devices activity monitoring are different in terms of the sensors used and inferred activities (Chen et al., 2008). The first ubiquitous device for physical activity monitoring is the pedometer. Pedometers are a type of motion sensor that monitors human activity as a person walks or runs. It consists of a sensor, such as a mechanical arm, magnetic switch, or an accelerometer, and software that typically measuring the number of steps an individual takes in a continuous manner (Crouter et al., 2003). Pedometers are typically light, portable devices that are worn on the hip or ankle. Correct placement is critical for pedometers to take precise measurement. However, recent advances have made them more applicable to other positions and orientations on the body [1].

More sophisticated devices exist commercially for activity monitoring. For instance, both FitBit and Phillips Tracmor devices incorporate multi-axis accelerometers to provide a convenient (orientation agnostic) method to infer calories burned. Impact Sports' ePulse monitor uses a heart rate and BodyMedia's GoWear unit combines four sensors (accelerometer, heat flux, galvanic skin response, and skin temperature) for this same purpose. Although these commercial offerings are widely available and fairly convenient, they only provide coarse activity information (step count, calories burned, distance traveled).

On the other hand, custom hardware are usually implemented using Micro-electro mechanical Sensors (MEMS). For example, Barralon et al. (Barralon et al., 2006a) work describes a MEMS architecture shows the results of the time spent in three postural state (lying, sitting, standing) and the periods of walking in a eHealth scenario using an unique accelerometer, placed on the

[1] http://www.omronhealthcare.com/

patients chest. The study determines the global position of the patients of the sensor wearer; they calculate the position of the patient considering the inclination of the sensor in every axis and then quantify this value. Finally, the study was made to evaluate the health of the patient and they obtain about 76% of accuracy rate.

Moreover, Bao et al. describes an architecture (Bao and Intille, 2004) to acquired human motion using five biaxial accelerometers worn on different parts of the body from 20 subjects. Extracted features from each accelerometer were: signal mean, energy, frequency-domain entropy, and correlation of acceleration and subsequently classify using a Decision tree, obtaining an overall accuracy rate of 84%. Although they reach a good accuracy, this architecture presents a big problem, to wear five devices over the body.

1.2.4.2 New sensors: Smartphones

Recently, research efforts have been made to develop systems to support smartphone sensing in different areas such as environmental monitoring, social networking, healthcare, transportation, etc. (Lane et al., 2010). Smartphones have evolved from communication devices to perceptive devices capable of inferring context around them. Nowadays smartphones hosts a GPS, gyroscope accelerometer, proximity and light sensors, high quality microphones and cameras (front and back), in addition to a range of communication interfaces, such as Wi-Fi, Bluetooth 4.0, Network communication (4G/LTE), and a near-field communication (NFC) interface.

Multi-core processors and gigabytes of memory allow smartphones to locally handle a large amount of data coming from these sensors and extract meaningful user information descriptors. In addition, the high and growing number of users of these devices maximizes the number of observed individuals, and may reach millions of devices distributed throughout the world. The participation of people becomes simpler, with no need of carrying specific equipment since participants already use their personal devices daily.

Due to its multiple communication technologies (GPRS, HSPA+, LTE, Bluetooth), these devices are able to send the collected data in a simple way, resulting in lower costs compared to other equipment. Projects developed previously, using mobile sensors applied in cars, use unsecured Wi-Fi access points for data transmission. Most of the services available on smartphones require an Internet connection, and most are massively used (e.g. social networking and email), creating communication opportunities for sending collected data.

The approach based on personal mobile devices maximizes the amount of collected data with no extra costs, neither need for maintenance of the devices. Moreover, sensing tasks are not restricted to a pre-defined time or to the lifetime of fixed sensors. For these reasons, the

focus of research has been driven up to mobile sensing, using and exploiting these devices' capabilities.

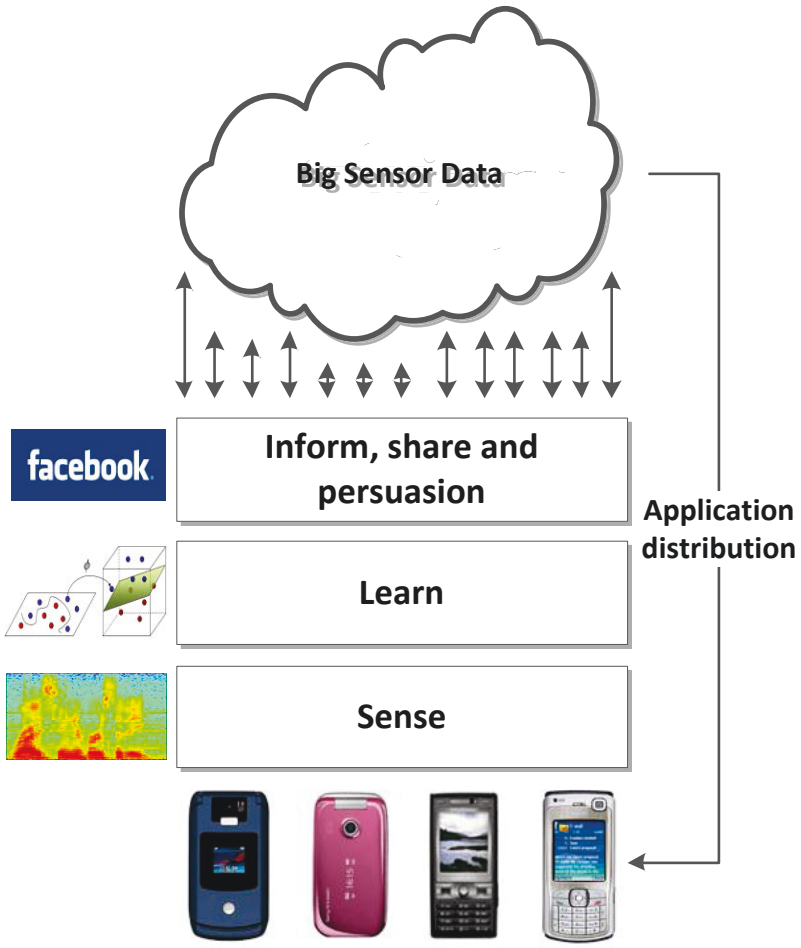

Figure 1.3: Cenceme architecture overview.

By embedding sensors to a mobile phone, mobile sensing provides the opportunity to track dynamic information about environmental impacts and develop maps and understand patterns of human movement, traffic, and air pollution (Rana et al., 2010). Researchers and engineers are making efforts in developing new more powerful and constructive applications.

One of the most notable contributions is presented up to now in mobile phone activity recognition is call Cenceme (Miluzzo et al., 2008). Cenceme is a personal sensing system which enables members of social networks to share their sensing presence with their relationships in a secure manner. Although, Cenceme does not use Social Networks Sites to collect information

(they only use in-built sensors) they introduce Social Network Sites into the activity recognition field by sharing user activity on Facebook.

Cenceme is centered on capture three different user's status: human activity (e.g., sitting, walking, driving), disposition (e.g., happy, sad, doing OK), habits (e.g., at the gym, coffee shop today, at work) and surroundings (e.g., noisy, hot, bright, high ozone). Although, in terms of the physical separation of functionality, the CenceMe architecture can be separated into two: back-end servers and user mobile device, the CenceMe proposed architecture is split in three different layers (Sense, learn and share):

- They perceive the mobile phone as a sensor, so, this layer is completely implemented in the smartphone. The principal goal of this layer is to collect user raw sensor data from the mobile phone. The in-built smartphone sensors used in this case are microphone to track the ambient noise, accelerometer in case of tracking people movements.
- In learn layer, they propose to use a variety of data mining techniques to infer user actions. These techniques are used to interpret embedded sensors raw data extracted in the previous layer. Depending on the user information to infer (sound, activities, etc.) they implement different techniques.
- The last layer and the most revolutionary idea of the CenCeme application is the share layer. At this point, they have composed a user profile inferring user action using machine learning techniques, and allow to share this information on Facebook. Their approach aims to share user context information in a web portal or SNS where sensor data and inferences are easily displayed.

Yohan Chon et al. (Chon and Cha, 2011) presents LifeMap architecture, an Smartphone-based Context Provider for Location-based Services. Authors split their architecture in four different components:

- All the sensors are placed on the low level, this level sends the obtained information to the Component Manager where information is processed and provide high-level information. Using high-level information from the Component Manager.
- The Context Generator generates a Point Of Interest (POI) which contains the user context. The context map is stored in a database to match and aggregate user contexts.
- And finally, the Database adapter is an interface to provide user context to other applications.

1.3 Recognition of Individual Emotions

The question of how humans perceive emotions has become central for the researchers of affective computing (Cowie et al., 2001). Emotions are fundamental to human experience, perception, and everyday tasks such as learning, communication, and even rational decision-making. Human emotion sensing may be obtained from a wide range of behavioral cues, gestures, facial expression, movements, speech or physiological signals (heart rate, salivation, ...). However, not of those are able to implement in a smartphone. In this section, we will focus on research works that face to these using available smartphone sensors.

1.3.1 Emotion Definition

This section does not aim to be an overview of the vast existing literature on emotion theory. Its goal is to present an overview of the definition and categorization of emotions. In the literature Affective state and Emotions are terms often used interchangeably, although affective stated describes a broader class of states than the term emotion. There are plenty of works trying to define what affective state is involved. In general, emotions are a short-term feeling (seconds/minutes), whereas moods are long-term (few days), and personalities are very long-term (months, years or even a lifetime) (Jenkins et al., 1998).

Although, the definition and categorization of affective state is one of the most discussed issues in the emotion literature, there is still no consensus. The first definition of emotion was given for Charles Darwin in the context of evolutionary theory as an action that is beneficial to evolution. Darwin's definition is still in use in classic psychological researchers. In the literature, classic psychological research supports the existence of a small, discrete number of emotions, that they are interconnected to our brain. The six basic emotions can be recognized universally are: happiness, sadness, surprise, fear, anger and disgust (Ekman and Friesen, 1978). On the other hand, some other researchers claim that some of these emotions do not refer to a real emotion. For instance, surprise is just an affective neutral state; therefore is not an emotion.

1.3.2 Emotion Recognition Techniques

Every technique use many different methods to process raw sensor to an emotion. However, the main steps can be categorized as pre-processing, segmentation, feature extraction, dimensionality reduction and classification.

1.3.2.1 Sensory Based Techniques

Face plays an important role in human communication process. Although humans detect and analyze faces and facial expressions easily, it is highly difficult to develop an computer automated system to perform this task (Fasel and Luettin, 2003). Most works in the literature tackle the problem of analyzing facial expressions based on Ekman and Friesen's work in 1978 (Ekman and Friesen, 1978). They described the *Facial Action Coding System (FACS)* concept where it is defined *action units (AUs)* as the causes of facial movement. Every AU is a set of diverse facial muscles that generate a facial action by their movement. Automatic facial expression analysis normally includes these three main processes:

- Face Acquisition: Includes eyes, eye-brows, mouth and nose recognition, but they are not restricted to other parts.
- Facial Feature Extraction: The extracted features can be analyzed statically or dynamically by monitoring and measuring the features variation in time.
- Facial Expression Classification: Using data mining techniques such us Neural networks or Hidden Markov Model.

Research	**Technique**	**Emotions**	**Rate**
(Essa and Pentland, 1997)	SVN	Surpise Joy	98%
(Anderson and McOwan, 2006)	SVM	Happy Surprised Sadness Disgust Fear Anger	86%
(Picard et al., 2001)	Neurofuzzy	Happiness Surprised Sadness Disgust Fear Anger	78%

Table 1.1: Review of the emotion recognition systems from Facial Expression.

The speech is the fastest and the most natural method of communication between humans. Emotional speech recognition aims to automatically identify affective State of a human being from his or her voice (El Ayadi et al., 2011). Speech expresses user affection through two ways: (1) explicit (linguistic) messages, and (2) implicit (paralinguistic) messages. Normally,

emotional speech techniques are focus on paralinguistic methods which reflects the way the words are spoken and linguistic techniques are used by Natural language processing researches.

On the other hand linguistic techniques are used by Natural language processing researches. Related work in human emotional speech recognition can be grouped according the types of speech features: Continuous features, qualitative features, spectral features, and TEO (Teager energy operator)-based features. Every emotion provokes a different spectral speech features. For example, the resulting speech of Joy, Anger, and Fear emotions is loud, fast and enunciated with strong high-frequency energy. On the other hand, sadness, producing speech that is slow, low-pitched, and with little high-frequency energy (Cairns and Hansen, 1994).

While recognize emotion through face-to-face channels is very easy, computer mediated communication may be cause confusion due to understand nuances of the expressions, jokes, sarcasm, etc. However, sometimes people makes up this lack, using emoticons. Some recent works notes that emoticons can provide emotion Information and improve CMC (Derks et al., 2008). Emoticons are described as graphic representations of facial expressions that are included in electronic messages.

Research	**Classifier**	**Emotions**	**Rate**
(Nakatsu et al., 1999)	NN	Neutral Happiness Surprised Sadness Disgust Fear Anger Playfulness	57,6%
(Lee et al., 2006)	Fuzzy SVN	Neutral Joy Sadness Anger Annoyance	73%
(Grimm et al., 2007)	ANN	Happy Angry Neutral Sad	83.5%

Table 1.2: Review of the emotion recognition systems from speech.

Darwin in 1872 (Darwin et al., 2002) described in detail the bodily expressions associated with emotions in animals and humans. Also, he proposed some principles underlying the organization of these expressions. Body gesture systems aim to recognize the emotional state of a person from the posture, the movements of the hands or arms. In general, the body

and hand gestures are much more varied than face gestures. Hence, there are many body movement combinations to express every single emotion. The existing approaches for body gesture recognition and analysis of human motion in general can be classified into three major categories:

- Model-based: consist on modeling the body parts or recovering 3D configuration of articulated body parts.
- Appearance-based: based on two dimensional information such as color/gray scale images or body silhouettes and edges.
- Motion-based: using directly the motion information without any structural information about the physical body.

Research	Classifier	Emotions	Rate
(Bianchi-Berthouze and Kleinsmith, 2003)	CALM ANN	Angry Happy Sad	95,7%
(Castellano et al., 2007)	INN-DTW	Anger Joy Pleasure Sadness	61%
(Kapur et al., 2005)	Perceptron SVN	Anger Sad Joy Fear	92%

Table 1.3: Review of the emotion recognition systems from body gestures.

1.3.2.2 Natural language processing (NLP)

While express emotion through face-to-face channels is easy to recognize, Computer-Mediated Communication (CMC) may be cause confusion. To understand nuances of the expressions, jokes, detecting subjective opinion documents or expressions, non-verbal cues may be an arduous task for humans and an nearly impossible task for computers.

Identifying the expressed emotions in text is very challenging for at least two reasons (Lin et al., 2014). The first one is that emotions can be implicit by specific events or situations. In the next sentence *When I see a cop, no matter where I am or what I'm doing, I always feel like every law I've ever broken is stamped all over my body*, it is possible to infer that the person

is scared or fear. Second one, gathering distinction between different emotions purely on the basis of keywords can be very subtle.

Although there is not any standard emotion word hierarchy, focus on the related research about emotion recognition, normally emotion is expressed as joy, sadness, anger, surprise, hate, fear according to the Ekman six basic emotions (Ekman and Friesen, 1978). In the context of emotion detection NLP is normally based on finding certain predefined keywords as happy, sad, anger, etc. A little overview about NLP features extraction techniques is presented:

- Part-of-Speech (POS): In corpus linguistics, part-of-speech tagging is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on its definition, as well as its context. It is also called word class, a category into which words are placed according to the work they do in a sentence. Commonly, there are 8 parts of speech (or word classes) and they are divided into two groups:
  - Open classes: nouns, verbs, adjectives, and adverbs.
  - Closed classes: pronouns, prepositions, conjunctions, and interjections.

  A common way to classify using POS features is reduced to calculate the percentage of words belonging to each POS in a tweet.

- LIWC Dictionary[2]: Linguistic Inquiry and Word Count3 (LIWC) is a text analysis software which provides a dictionary covering about 4,500 words and word stems from more than 70 categories. The software is available in 11 languages (Spanish is included). In this case, the classification method counted the number of positive/negative words based on the set of collected emotion words, and used the percentage of words that are positive and that are negative as features.

- Adjectives: In sentiment analysis, adjectives are usually considered as effective features since they can be good indicators of sentiment. Some research (Pang et al., 2002) shows that using adjectives alone produces competitive results with those obtained by using n-grams in sentiment classification of movie reviews. In order to classify each tweet adjective is included in a feature vector.

- Emoticons: Other way to face NLP is rely on the used emoticons. Some recent work, however, notes that emoticons can provide emotion information and improve CMC (Derks et al., 2008). Emoticons are described as graphic representations of facial expressions that are included in electronic messages.

[2]http://www.liwc.net/

- N-grams: In the fields of computational linguistics and probability, an n-gram is a contiguous sequence of n items from a given sequence of text or speech. An n-gram could be any combination of letters. However, the items in question can be phonemes, syllables and letters, although using words give more information to the developer.

Other way to face NLP is rely on the used emoticons. Some recent work, however, notes that emoticons can provide emotion information and improve CMC (Derks et al., 2008). Emoticons are described as graphic representations of facial expressions that are included in electronic messages.

1.3.2.3 Multimodal System

Finally, multimodal system combines human emotions observations from more than one technique in order to provide a robust and complete description of the human emotional state. Depending on the context, there are situations where is impossible to capture human emotions with a concrete technique. For example, in noisy environments speech emotion recognition is hardly difficult as well as in a dark place try to capture body gestures o facial expressions. Hence, a good emotion recognition system depends on access to more than one technique. Table 1.4 shows the classification rate in multimodal architectures is higher than single ones. Besides, the number of recognition techniques is also higher. Hence, multimodal systems present more advantages, precision and number of emotions over single ones.

1.3.3 Emotion representation and EML

As well as the emotion does not have a commonly agreed theoretical definition; a categorization or representation model there is no consensus. Nowadays, there exist two different representation models: Categorical and dimensional. Categorical model of emotion has its roots in the evolutionary theories which claims that emotions are biologically determined, discrete and belong to one of a few groups. These groups are consider fundamental or *basic*. However, the problem is that which emotions are considered basic. In this case, according with (Ekman and Friesen, 1978) definition of affective state the basic emotions are normally considered: happiness, sadness, surprise, fear, anger and disgust.

This view presents reduce sharply the number of emotions. Some researchers think that any basic emotion may be decomposed into secondary emotions. This process is very similar to the way that any color is a combination of some basic colors. Emotions is by mixing and matching the basic emotional labels as if in a palette of primary colors *Palette theory* as figure 1.4 shows.

Research	Techniques	Emotions	Rate
(Cowie et al., 2001)	Voice Face Speech	Neutral Afraid Disgusted Joy Sad Angry Surprise	98%
(Gunes and Piccardi, 2007)	Face Gestures	Disgust Happiness Fear Anger Uncertainty Anxiety	91%
(De Silva and Ng, 2000)	Face Speech	Neutral Hate Anger Grief Joy Reverence	72%

Table 1.4: Review of emotion recognition systems from Multisensor systems.

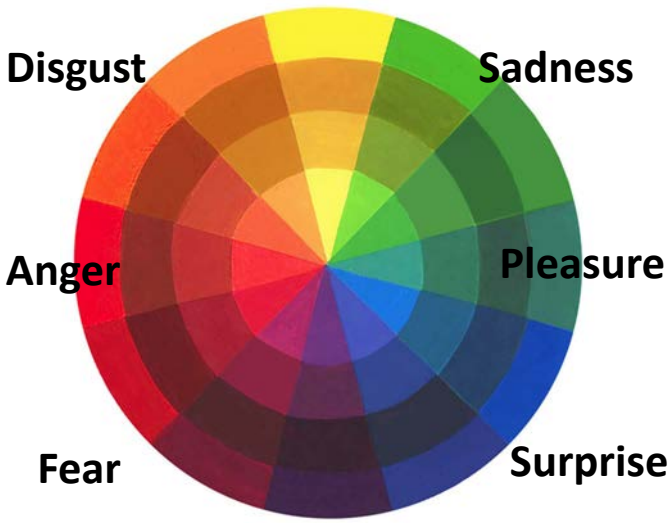

Figure 1.4: Emotion representation: Categorical model.

In contrast to categorical model, dimensional models do not fix a finite set of emotions. Alternately, they attempt to find a finite set of underlying features into which emotions can be decomposed, any combination of features give a different affective state. Under this model,

emotions are described in terms of three components or dimensions (Schlosberg, 1954). The first dimension aims to describe the degree of pleasantness underlying the emotional experience. The second one describes the level of activation of the emotion and finally the last one defines the level of attention or rejection.

The three dimension approach is synthesized in figure 1.5 where a concrete emotion (e) is the result of the intersection between every different dimensions (d) whose values are determined by pattern of signals (s).

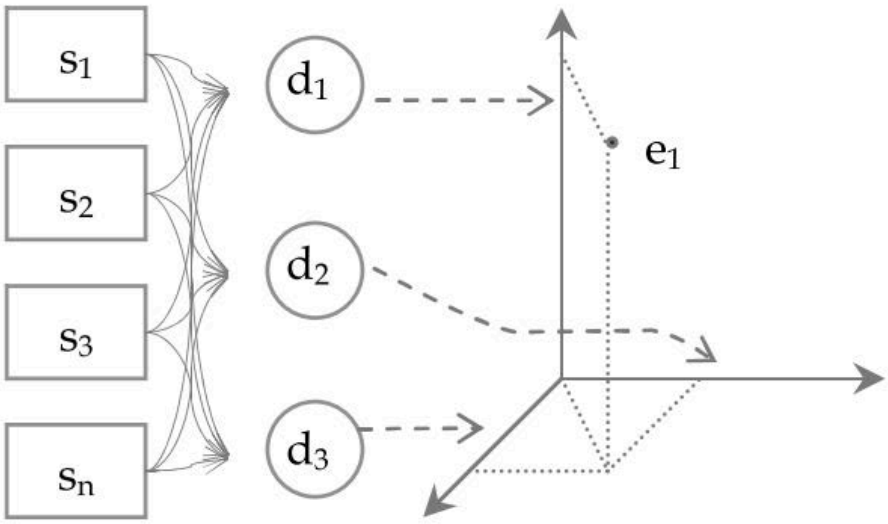

Figure 1.5: Emotion representation: Dimensional model.

In 2005, thanks to the necessity of a standardized way of representing emotions emerged the W3C Emotion Incubator group which describes the foundations for such a emotion standard language (Emotion Markup Language) [3]. Emotion terms are not consistent in the literature, for that reason, EML has a glossary in order to reduce ambiguity by describing emotions.

Although several non-standard markup languages referring to emotion have been proposed previously, emotion researchers have studied none of those. These languages has been designed to ad-hoc platform instead of use it in a broad range of application areas. Those solutions reduce its model to the basic emotions (anger, fear, joy, and sadness) and it is out of scope other possible solution to model emotions (EARL). The EARL is a syntactically simple XML language designed specifically for the task of representing emotions and related information in technological contexts (Schroder et al., 2006).

In order to test if the researchers requirements were successfully complete, they tested with

[3]W3C EMLIncubator Group http://www.w3.org/2005/Incubator/emotion/

39 use case scenarios grouped into three different areas (Manual annotation of data, Automatic recognition of emotion-related states from user behavior and Generation of emotion-related system behavior). Emotion terms are not consistent in the literature, for that reason, EML has a glossary in order to reduce ambiguity by describing emotions:

- Action tendency: Action tendencies can be viewed as a link between the outcome of an appraisal process and actual actions.
- Affect/Affective state: This term is consider it to be more generic than "emotion", in the sense that it covers both acute and long-term, specific and unspecific states. The term "affective state" is used interchangeably with "emotion-related state".
- Appraisal: The term "appraisal" is used in the scientific literature to describe the evaluation process leading to an emotional response.
- Emotion: is used in a very broad sense, covering both intense and weak states, short and long term, with and without event focus.
- Emotion-related state: The main concept of the EmotionML. It is a term for the broad range of phenomena intended to be covered by this specification. The final report of the Emotion Incubator Group has a section where is described all the emotion-related state terms (e.g. Ekman's "big six", OCC categories, etc.)
- Emotion dimensions: A small number of continuous scales describing the most basic properties of an emotion. According to three dimensions scale: valence (sometimes named pleasure), arousal (or activity/activation), and potency (sometimes called control, power or dominance) or just two dimensions described below.
- Full-blown emotion: Intense states with a strong focus on current events.

1.4 Multi-Sensor Frameworks

Data fusion is a multi-disciplinary research area borrowing ideas from many diverse fields such as signal processing, information theory, statistical estimation and inference, and artificial intelligence. Normally, multisensory data fusion has been applied in the Department of Defense (DoD) areas such as automated target recognition, battlefield surveillance, and guidance and control of autonomous vehicles. However, nowadays, it is also widely applied to non-DoD applications like user monitoring, medical diagnosis, and smart buildings, to name a few (Esteban et al., 2005).

This new wave has been impulse by huge evolution of monitoring applications. The large number of new sources of user information opens up new possibilities in information fusion research field. Multisensor data fusion is a technology to enable combining information from several sources in order to create a unified picture. Generally, performing data fusion has several advantages (Khaleghi et al., 2011). These advantages mainly involve enhancements in data authenticity, availability or accuracy. In the next subsections, the most relevant architecture and models are briefly described in the information fusion research field (Galeana-Zapién et al., 2014).

1.4.1 JDL model

Various conceptualizations of the fusion process exist in the literature, although, the most common and popular conceptualization of fusion systems is the JDL model (Hall and Llinas, 1997) developed for military applications. The military community has developed a layout of functional architectures based on the Joint Directors of Laboratories (JDL) model for multisensor systems. The original JDL model considers the fusion process in four increasing levels of abstraction, namely, object, situation, impact, and process refinement 1.6. Despite its popularity, the JDL model has many shortcomings, such as being too restrictive and especially tuned to military applications.

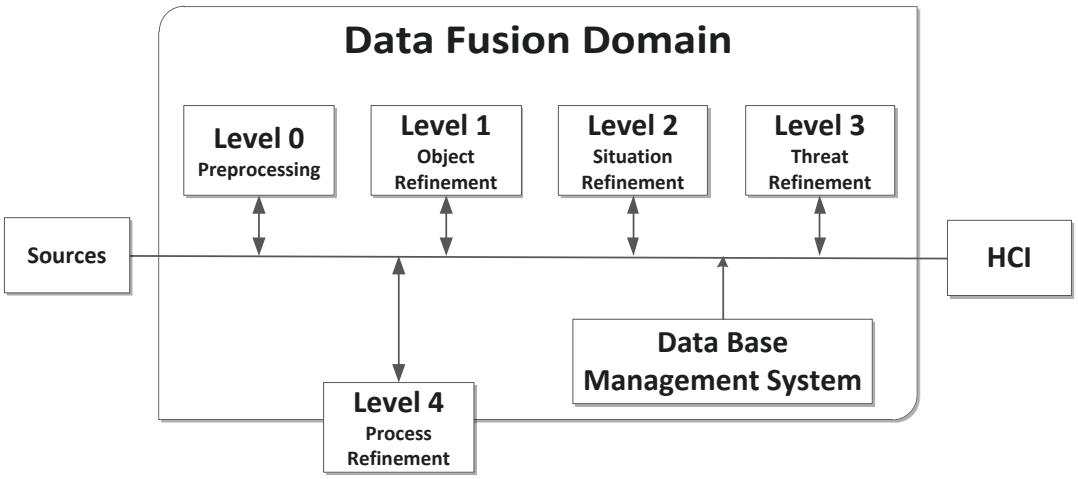

Figure 1.6: JDL Information Fusion model.

The JDL model was never intended to decide a concrete order on the data fusion levels. Every level is not exempted to be processed consecutively and also be executed concurrently.

- Level 0, *sub-object data assessment*: is generally aimed at combining signal level data to obtain initial information about an observed target's characteristics. Normally, is associated with pre-detection activities such as pixel or signal processing, spatial or temporal registration.

- Level 1, *object refinement*: At this level, to identify and locate objects is attempted. Hence, it is reported the object situation by fusing the attributes from diverse sources. The steps included at this stage are:

  - Alignment: Processing of sensor measurement to achieve common time base and a common spatial reference.
  - Association: A process by which the closeness of sensor measurement is completed.
  - Correlation: A decision-making process which employs an association technique as a basis for allocation sensor measurement to the fixed or tracked location of an entity.
  - Correlator-tracker: A process which generally employs both correlation and fusion component processes to transform sensor measurements into states and covariance for entity track.
  - Classification: A process by which some level of identity an entity is established either as a member of a class, a type within a class or a specific unit within a type.

- Level 2, *situation assessment*: Attempts to construct a picture from incomplete information provided by level 1, that is, to relate the reconstructed entity with an observed event. Entities are associated with environmental, doctrinal and performance data. These models are then compared with previously learned models in order to develop a pattern and predict next steps. This level is the most important level in data fusion as it helps the next level process.

- Level 3, *threat assessment*: Interprets the results from level 2 in terms of the possible opportunities for operation. The impact of the situation is estimated, analyzing pros and cons of taking one action over another one, and the underlying cost and measures are computed which helps in estimating the usage of resources.

- Level 4, *process refinement*: Process refinement is an element of resource management and used to close the loop by re-tasking resources (e.g. sensors, communications and processing) in order to support the objectives.

The JDL model has been very popular for fusion systems, although presents some drawbacks such as being too restrictive and especially tuned to military applications which have been the

subject of several extension proposals (Steinberg et al., 1999) attempting to alleviate them. The JDL formalization is focused on data (input/output) rather than processing. Despite its popularity, the JDL model has many shortcomings:

- It is a data-centered or information-centered model which makes it difficult to extend or reuse applications built with this model.
- The model is very abstract, and complicates to properly interpret its parts and to appropriately apply it to specific problems.
- Although the model is helpful for common understanding, it is not a developer guide to identify those methods that should be used. Hence, the model does not help in developing an architecture for a real system.

1.4.2 Waterfall model

The waterfall information fusion model was proposed by Markin et al. (Markin et al., 1997) (Figure 1.7 shows A representation of the waterfall model). It has been used in the defense data fusion community in Great Britain, but it has not been significantly adopted elsewhere.

This architecture emphasizes on the processing functions on the lower levels. It can be seen from this figure that the data flow operates from the data level (lower level) to the decision-making level (higher level). The sensor system is continuously updated with feedback information arriving from the decision-making module. The feedback element advises the multi-sensor system on re-calibration, re-configuration and data gathering aspects. There are three levels of representation in the waterfall architecture and they are described as follows:

- At level 1, the raw data is properly transformed in order to provide congruent information to the system. This task is performed studying the models of sensors and the models of the measured phenomena. These models could be based whether on experimental analysis or on physical laws. This level correspond to the level 0 in the Joint Directors of Laboratories architecture.
- At level 2, the required features are extracted from the raw data from the previous level and then fusion is performed on those features. The aim of this level is to improve the information delivered while minimizing the data content, thus achieving symbolic value of the object. The end result of this level is the collection of estimates along with the probability and beliefs associated with them. Level 1 in the JDL architecture matches with the second level in the waterfall architecture.

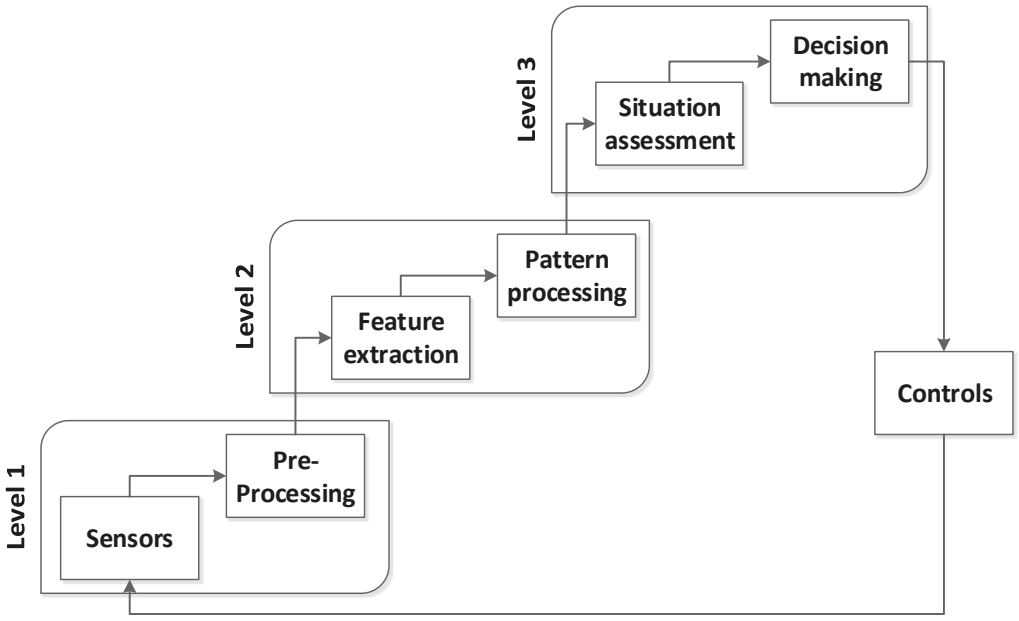

Figure 1.7: Waterfall Information Fusion model.

- At level 3, the information available is measured to obtain the possible outcomes of the situation relates objects to events. Possible routes of action are assembled according to the information that has been gathered, the libraries and databases available, and the human interaction. The decision making level performs the same operations that level 2 and 3 in the JDL model.

Although, waterfall model is more accurate in analyzing the fusion process than other information fusion models, it suffers from the same drawbacks to the JDL model. This system is continuously updated by the information obtained from a decision making model. However, waterfall model omits any feedback data flow instead of JDL model which every level is interconnected.

1.4.3 OGC: Sensor Web Enablement

The OGC provides a large number of specifications; among them we can find the Sensor Web Enablement (SWE) family of standards. This is essentially a set of standards that allow for exposing sensor networks as Web Services making them accessible via the Web (Clarke et al.,

2009). The SWE framework is composed of seven standards; the OGC members have approved four of them as official standards.

The SWE initiative has developed draft specifications for modeling sensors and sensor systems (SensorML, TransducerML), observations from such systems (Observations and Measurements) and processing chains to process observations (SensorML). The draft specifications provide semantics for constructing machine-readable descriptions of data, encoding and values, and are designed to improve prospects for plug and play sensors, data fusion, common data processing engines, automated discovery of sensors, and utilization of sensor data.

- SensorML: it is a language based on XML schema to describe the sensor systems. It encodes a lot of features for sensors, such as discovery, geolocation processing observations, mechanisms for sensor programming, and subscriptions to sensor alerts. In particular, it provides standard models and XML schemes to describe processes, and instructions for obtaining information from observations. SensorML enables discovery, access and query execution for the processes and sensors it models.

- Observations and Measurements (O & M): this model in particular is featured in the SOS specification, coupled with an XML encoding for observations and measurements originating from sensors, and archived in real-time. It provides standardized methods for accessing and exchanging observations, alleviating the need to support a wide range of sensor-specific and community specific data formats.

- Sensor Observation Service (SOS): it corresponds to the Observation Agent specified in the previous section. This is the service responsible of the transmission of measured observations, from sensors to a client, in a standard way that is consistent for all sensor systems including remote, in-situ, fixed and mobile sensors. It allows the customer to control the measurement retrieval process. This is the intermediary between a client and an observation or near real-time sensor repository.

- Sensor Planning Service (SPS): it corresponds to the Planning Agent.

The benefits of exposing sensor nodes and sensor networks as service providers in a generalized SOA motivated the emergence of the Sensor Web Enablement (SWE) activity by the Open Geospatial Consortium.While SWE work is a valuable starting point, it is too generic to be directly applicable, and requires tailoring to be adapted to the building and facilities management domains despite its benefits. The main advantages that preset SWE approach are abstraction, separation of operations in reusable objects and applies them to the other

environments, also taking into account the particular domain's inherent characteristics. There are plenty of works using SWE as a framework to store sensor information, for example in (Fattoruso et al., 2015) is used to measure the quality of the water.

2

Modelling and Processing People Context in a Mobile Scenario

SOME challenges are presented from many perspectives: the real availability of the foreseen technology, its integration in people lives and finally, the acceptability of the resulting services by the potential users. There are several basis concepts to consider for retrieving people context, however, the most important is that sensors devices must be collect user context information in a unobtrusive way. Many technologies are conceived as hand-held or wearable, taking advantage of the intelligence embedded in the environment.

The chapter begins by exploring the sources of information context behind this proposal (Intelligent mobile devices and Social Networks Sites) and introducing the main technological aspects involved in collecting and managing the context. Besides, this chapter introduces a review of the available sources of context in a mobile scenario. Nevertheless, the description of our architecture is considered the main goal of this chapter. The architecture, called inContexto, is layered in 4 levels where the user information evolves from the collected raw sensory data to high-level user context information. inContexto architecture aims to describe how best to perform this task in order to offer to the users new and amazing services. The research to date has presented some drawbacks:

- User context information is mainly related to location which is valuable in many areas, including security, RF coverage assessment but do not contextualizes all user context information.
- Current frameworks are uniquely focused on retrieving and storing user context information, instead of completing it from other sources or using machine learning techniques to infer new context information. Current frameworks just put effort in storing this information.

- Most of these frameworks are commercial systems and they do not provide Application Programming Interface (API) in order to allow third party applications to store and retrieve user context information.

2.1 Context Evolution: From raw data to Action Context

The main content of the chapter is developed in this section. It goes with a description of artificial intelligence algorithms and techniques for automatically inferring people context information on a mobile device based on the retrieved information about the user of current real-time acceleration data measured and information posted on the Social Networks Sites.

One of the main line of research in user context recognition research field is to recognize inconspicuously activity (physical or mental) of individuals and react to their needs. There are many different methods to retrieve user context information from sensing the literature, however, most of them are develop without taking into account the user convenience. Since, activity recognition can be faced as a Pattern recognition problem, the principal steps are categorized as: data acquisition, pre-processing, segmentation, feature extraction, dimensionality reduction and classification (Krishnan et al., 2009).

In our proposal, people context is abstracted as a Context Pyramid (Figure 2.2). Raw data information from diverse sensors is the foundation of the Context Pyramid and the top layer is the high level action (Action Context) where information is more specific. Based on the Raw Sensor Data, we can extract physical and emotional parameters such as position coordinates, acceleration, heading, angular velocity, velocity, and orientation. Features/Patterns of physical parameters are generated for further pattern recognition, which generate the simple context such as location, motion, emotional state and surroundings. Activity-Level Descriptors combine the simple contextual information into the activity level. On the top of the pyramid, high-level context includes rich social and psychological contexts, which is basically expressed in natural language.

Normally, low-level sensor fusion not often requires sophisticate knowledge representation mechanisms. However, this interest is shifting to high-level context information, which needs expressive and interpretable representation and reasoning formalisms for situation assessment and impact evaluation.

The proposed model (Context Pyramid) aims to identify which information is represented in each one of the successive levels that raw data go through to become high level user information, in consonance with the High-level context abstractions. The pyramid represents the context

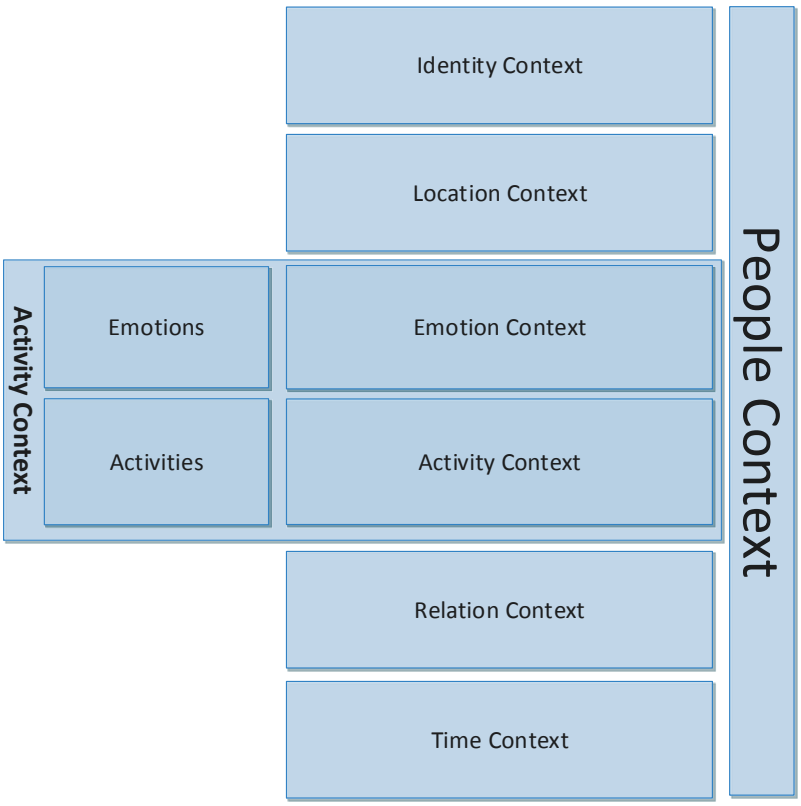

Figure 2.1: Representation of an Entity in mobile environments.

evolution from the bottom of the pyramid (raw data) to the top level (high level action or Action Context). This evolution, from raw sensor data to Action Context, is described based on Dey context definition: Status (User emotion and Physical actions), Social, Location and Identify. The following subsections describe the evolution from raw sensor data to Action Context of each concrete context according Dey definition: Status (User emotion and Physical actions), Social, Location and Identification.

2.1.1 Level 0: Raw Data and Features

Level 0 aims to collect and pre-process raw data from different sensors. Besides, it is extracted those features which well represented the user actions. Features can be defined as the abstractions of raw data. The raw sensor data acquired by phones, independent of the amount or source (e.g., accelerometer, camera), are worthless without interpretation. Normally, low-level sensor fusion not often requires sophisticate knowledge representation mechanisms. However, this interest is shifting to high-level context information, which needs expressive and interpretable

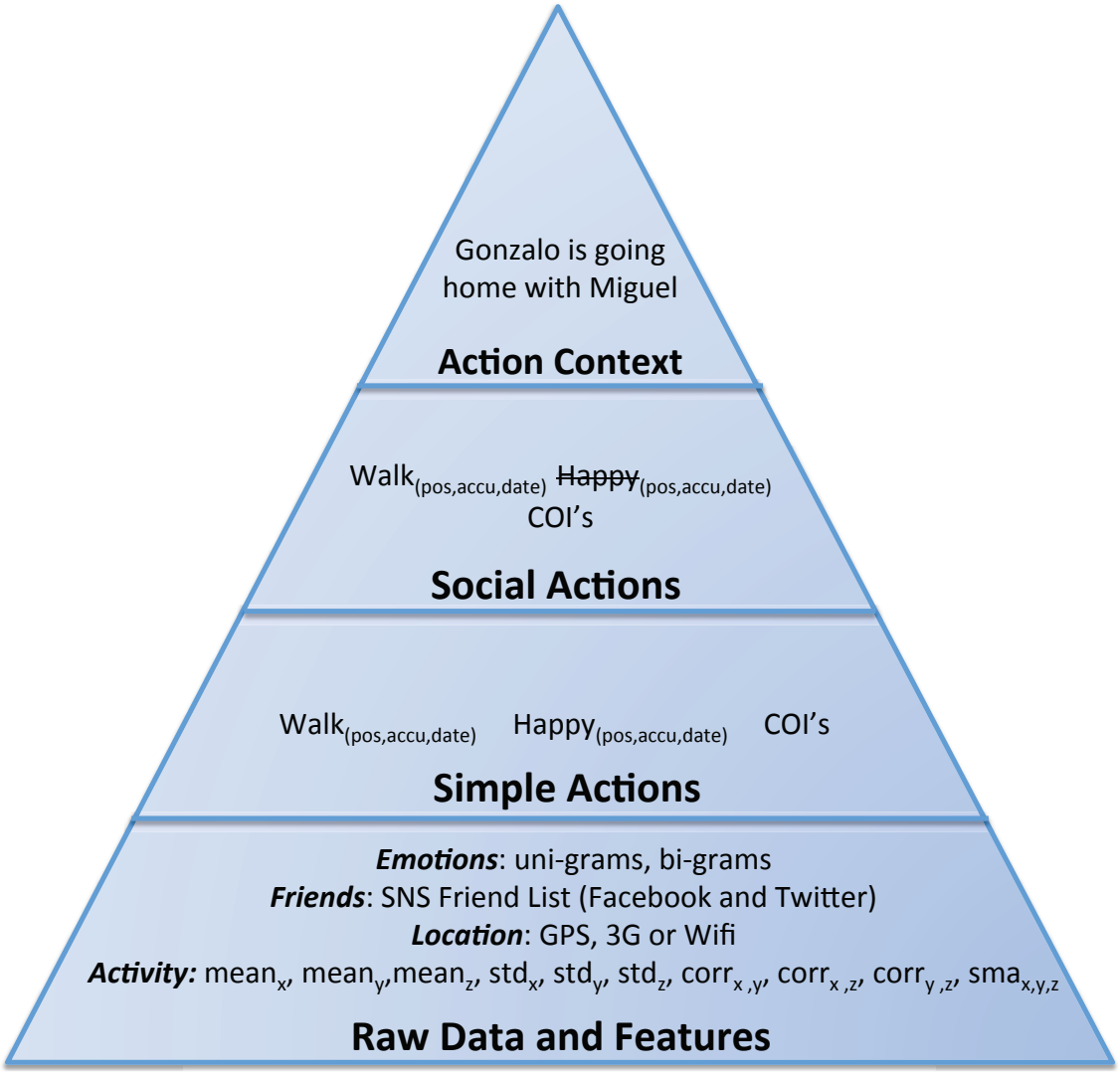

Figure 2.2: People context information evolution from raw data to high level actions according to the proposed model (Context Pyramid).

representation and reasoning formalisms for situation assessment and impact evaluation.

The objective of feature extraction is to represent an activity with the main characteristics of a data segment. Therefore, this level aims to process and select which features are better to identify entity context. The module processes several sensor observations into a vector features that help to discriminate between user statuses.

2.1.1.1 Entity Identity

The first challenge in context sensing applications is the identification of those modalities of raw sensory data that are the most descriptive of the people status and their surroundings. Although it seems obvious to comment that identification in a context application is crucial, not only is important to determine who is doing an activity. A good identification opens a wide range of new possibilities in order to offer new context based services or improve existing ones. For example, daily users perform the same activities, if the user identification is performing correctly, the system can store these actions, creating a user activity log. Afterwards, it may anticipate user movement offering more adapted information. Hence, each user stored activity or status can be associated with another activity that the user may engage in.

The identification process may be implemented basically in two ways: via software or hardware. Hardware identification may be performed using a device terminal identification such as MAC address, telephone number, etc. Normally, such identification is very common in stationary or external devices, which are characterized by their durability over time and its low turnover rate. Inside this group of devices can point out as most important: Zigbee devices, meteorological stations, Wi-Fi access points, etc. However, this method presents some drawbacks because of smartphone itself does not identify the concrete person who carries it. Hence, if another user manipulates the same device, there will be identification problems.

On the other hand software identification allows to identify uniquely the user who is on the scene. Software identification methods currently available are basically focus on two types: User credentials (user-name and password) and the delegation of the identification process to other applications, usually Social Networks Sites (Kaila, 2008). Facebook, Google, Twitter to name a few provides a system (OAuth or OpenID) that allows to any application identifying every user in theirs Social Network Site. In this way, to delegate the identification process to SNS is much safer than creating a new identification process. Besides, Users trusts in SNS it is a system that the user trusts. From the end user viewpoint these systems are almost transparent. The user will just be redirected to the identification system site and then redirected back after having successfully authenticated.

2.1.1.2 Activity Recognition: Raw Data and Features

A low-level sensing module continuously gathers relevant information about the user activities using available sensors. Commonly, cameras and microphones are probably the most used sensors to infer user context in AmI scenarios. However, these sensors present several issues such as Energy consumption and computational processing cost. New kind of sensors (Accelerometers,

Gyroscopes, Magnetometers, etc.), also known as Micro-Electro Mechanical Sensors (MEMS), are traditionally used to retrieve user movements. Basically, using this kind of sensors is possible to obtain basic actions taken by the user like running, walking, driving, etc.

In the literature, there are many different methods to retrieve user activity information from raw sensor data in the literature. However, the principal steps can be categorized as collecting, preprocessing, segmentation, feature extraction and finally classification (This step is described in level 1). First of all, collecting data depends on the activities to infer. Before to collect user information, to chose which sensors are the most suitable in order to infer the selected activities is extremely necessary. Besides, there is a great importance of taking into account sensor limitations, energy consumption, connectivity, frequency, etc. Every single sensor ($s_1, s_2 tos_n$) may provide a big amount of data ($rd1_1, rd1_2 tord1_n$).

Once we have collected raw data is necessary to preprocess this information in order to prepare the data for analysis. As well as collecting information there is not a specific technique to preprocess data. For example, to remove high frequency noises pikes using techniques like non-linear, low-pass median filters (Mathie et al., 2003), Laplacian filters (Bidargaddi et al., 2007) and Gaussian filters (Krishnan et al., 2008). Third, segmentation plays an important role in systems that perform continuous activity recognition.

The first challenge in activity recognition using mobile phone is the identification of those modalities of raw data that are the most descriptive of the concrete context to infer. Interdisciplinary efforts and domain knowledge are crucial in determine which techniques are the most suitable to infer people activities using inertial sensors (Tapia et al., 2007). Besides, for the inference to be made, first we need to decide which aspects of the raw sensor data are the most representative of the phenomena we would like to explain.

Although smartphones provides highly multimodal sensing, and also they are unobtrusively carried by their owners at all times normally, GPS, Accelerometer sensors or even together are the most used sensors to tackle the activity recognition problem. However, smartphone sensing applications are still presented the standard mobile sensing constraints to take into consideration. The principal drawback comes from the battery consumption.

It is possible reduce its energy consumption either by sampling less often or, in case it hosts multiple sensor types, by preferring low-power sensors to more power hungry ones. Both ways to reduce energy consumption present also drawbacks of either completely missing to sense important events or sensing them with an insufficient resolution. Although GPS and Accelerometer approaches are more accurate than accelerometer ones, normally, accelerometer based approaches are preferred over the other ones. Table 2.1 shows a summary of works that

has taken place in this space along with the types of activity modes inferred, the test user base, and the classification accuracy.

Research	Classes	Sensors	Mobile
Cenceme (Miluzzo et al., 2008)	Still Walk Run	Accelerometer	Yes
lifeMap (Chon, 2011)	Still Walk Motor	Accelerometer Magnetometer Wifi GPS	Yes
Borrielo (Lester et al., 2006)	Still Walk Stairs up and down Riding elevator Brushing	GSM Wifi	No

Table 2.1: Review of the activity recognition features using inertial sensors.

On the other hand, features can be extracted from a single sensor (accelerometer, GPS or even together). In many domains, however, certain features types have crystallized out as the most informative than others. In concrete, there are two types of extract features from accelerometer raw data. The first ones are those techniques, which use frequency properties analysis (DWT, CWT, and STFT), and secondly those that create a vector with statistical methods (SMA, signal mean, correlation, etc.). In terms of the frequency features there are a huge diversity of techniques to apply. However there are mainly three: Short Term Fourier Transform (STFT), Discrete Wavelet Transform (DWT) and Continuous Wavelet Transform (CWT). Table 2.2 lists the most commonly observed features in the activity recognition field using inertial sensors.

The basic idea behind of STFT is to apply Fast Fourier Transform (FFT) in sequence during short sampling periods. For each different period if time a different spectrum is obtained and the whole of such spectra indicates the time-frequency distribution (Gurley and Kareem, 1999). STFT is a technique capable of analyzing non-stationary signals and it is defined by equation 2.1.

$$STFT(t^1, f) = \int_t x(t) * e^{-2\pi i f t} * w(t - t^1) dt \tag{2.1}$$

Where $x(t)$ is the signal to analyse and w apodization function (Hanning).

A spectrogram is a time-varying spectral representation that shows how the spectral density of a signal varies with time. The active frequencies (Red and yellow bars) depending on the action taken (Steady actions or not steady action) are clearly distinguishable. Non-active frequencies are colored in blue which represent when the user is doing a sedentary action (Standing, sitting, etc.).

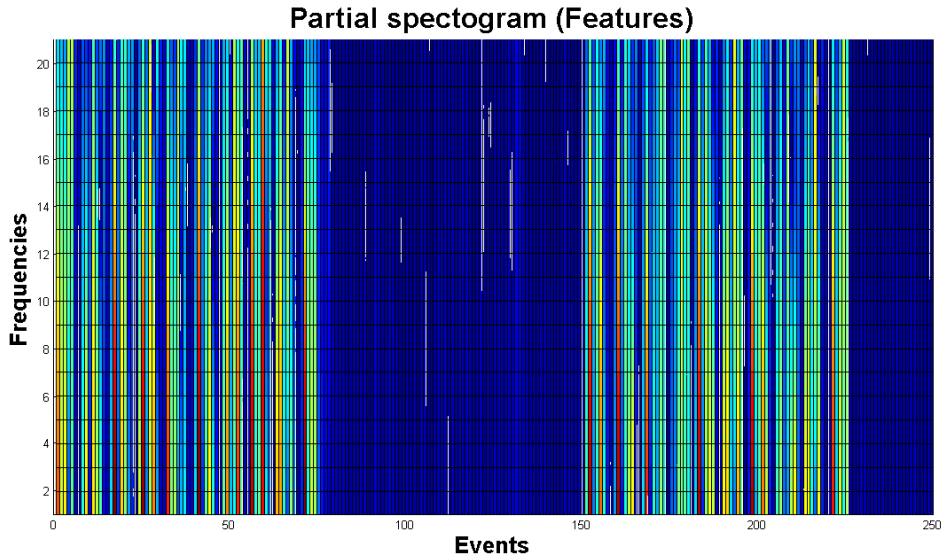

Figure 2.3: Activity Recognition Spectrogram Example.

Discrete Wavelet Transform is also considered an efficient technique or non-stationary signal analysis, providing representative characteristics of time and frequency (Barralon et al., 2006b). DWT is described by equation 2.2. DWT is based on the Mallat algorithm (Mallat, 1999), which uses a series of high (HPF) and low pass filters (LPF) to progressively find the wavelet coefficients.

$$DWT_{j,k} = \sum_{N} x(n)\overline{h_{j,k}(n)} \tag{2.2}$$

Where n represents the discrete time index, $x(n)$ is the discrete time original signal, $h(n)$ is the discrete time wavelet basis function, N is the total number of $x(n)$ samples, j is the time scaling, and k is the shifting of the discrete wavelet function $h(n)$ through the input signal $x(n)$.

Finally, while the continuous wavelet transform is the most compact and efficient, its power of two relationships in scale fixes its frequency resolution. Sometimes it is desired to achieve

smaller frequency bands than DWT allows. This is possible by using scales that are more closely spaced together than the 2^i relationship). CWT allows selecting the scale range. The number of coefficients necessary to describe the signal may be very much larger than the signal length, as the CWT oversamples the signal and wavelet coefficients contain partial redundancies of information. Moreover, CWT needs not information over the complete range of frequencies in the signal, which allow to the user select a very narrow range of a particular frequency band.

CWT is used to split a continuous-time signal into wavelets. Unlike Fourier transform, the CWT is able to construct a time-frequency representation of a signal, which offers very good time and frequency localization. The CWT (continuous wavelet transform) of a signal x(t) is depicted as follows:

$$X_w(a,b) = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} x(t)\psi^* \left(\frac{t-b}{a}\right) dt \tag{2.3}$$

In order to reconstruct the signal, we can use this inverse formula:

$$x(t) = \int_0^{\infty} \int_{-\infty}^{\infty} \frac{1}{a^2} X_w(a,b) \frac{1}{\sqrt{|a|}} \tilde{\psi} \left(\frac{t-b}{a}\right) db\ da \tag{2.4}$$

Where i corresponds to dilation, and j to translation. For a finite digitally sampled signal, the integral will be replaced with a summation, and the time t is replaced by the discrete n.

On the other hand, statistical or time domain features are used over the time in order to recognize people activity (Cleland et al., 2013). Time-domain features include basic waveform characteristics and signal statistics and they are directly derived from a data window. Normally, the most used characteristics are the following:

- *Mean*: The mean acceleration value of the three axis signal over a window is the DC component of the acceleration signal. In (Ravi et al., 2005) extract the mean value from each of the three axes of the accelerometer and present the accuracy of the mean value feature for classification.

- *Variance*: To compute the variance of a 3 accelerometer components (The square of the standard deviation Linear).

- *Correlation Coeffs*: Correlation between axis pair, XY, XZ, YZ. Correlation is calculated between each pair of axes as the ratio of the covariance and the product of the standard deviations as is shown in the equation 2.5. Correlation is especially useful for differentiating among activities that involve translation in just one dimension.

$$corr_{(x,y)} = \frac{cov(x,y)}{\sigma_x \sigma_y} \tag{2.5}$$

- *Signal Magnitude Area (SMA)*:SMA features are shown to be used effectively for identifying periods of daily activities. Moreover, it is possible to implement in order to identify static and dynamic activities. The SMA is equal to the sum of acceleration magnitude summations over three axes of each window normalized by the window length. The discrete form of the SMA can be given by

$$SMA = \frac{1}{w}\left(\sum_{i=i}^{w}|x_i| + \sum_{i=i}^{w}|y_i| + \sum_{i=i}^{w}|z_i|\right) \tag{2.6}$$

  where w is the window length; x_i, y_i and z_i represent ith components of the x-, y- and z-axis samples in a window.

- *Signal magnitude vector (SVM)*: SVM, defined in 2.7 essentially provides a measure of the degree of movement intensity.

$$SVN = \sqrt{x_i^2 + y_i^2 + z_i^2} \tag{2.7}$$

2.1.1.3 User Emotional Activity

By creating applications able to change its behaviour according to the human emotional state is an amazing task but its still in its infancy and of course it is not clear which sensors are the most suitable to face the problem unobtrusively (Cowie et al., 2001). Human emotional state may be obtained from a wide range of behavioural cues, which are determined, by gestures (facial expressions, head movements, Natural language processing, etc.), and speech (parameters such

Refernce	Features	Activities	Accuracy
(Bao and Intille, 2004)	Mean	Walking	Decision tree 84%
	Entropy	Sitting	kNN 83%
	Energy signal	Running	Naive Bayes 52%
		Cycling	
		Vacuuming	
		Folding laundry	
(Ravi et al., 2005)	Mean	Standing	Naive Bayes 64%
	Std	Walking	SVM 63%
	Energy signal	Running	Decision tree 57%
	Correlation	Stairs up & down	kNN 50%
		Vacuuming	
(Pirttikangas et al., 2006)	Mean	Typing	Neural network 93%
	Std	Watching TV	k-NN 90%
	Signal	Drinking	
		Stairs up & down	
(Miluzzo et al., 2008)	Mean	Still	JRIP 78%
	Std	Walk	
	Energy signal	Run	
(Barralon et al., 2006b)	STFT	Still	Threshold-based 69%
	DCT	Walk	
	CWT		
(Yang et al., 2008)	Mean	Running	Neural network 95%
	Correlation	Scrubbing	k-NN 87%
	Energy	Brushing teeth	
	Interquartile		
	RMS		
(Lee et al., 2003)	Running	FFT	Threshold-based 95%
		Stairs up & down	
		Walking	
		Sitting	
		Lying	
		Standing	

Table 2.2: Summary of notable works involving activity detection using inertial. The table includes the type of activities, the features used and detection accuracy achieved

as pitch, energy, frequency and duration), or physiological signals (heart rate, skin conductivity, salivation, etc.).

Psychological techniques shows acceptable performance but has a critical weakness that reduces its widespread use; they are obtrusive to users and need special equipment or devices. Recognizing user emotion from their psychological signals is necessary to wear specialized equipment such as a skin conductance sensor, blood pressure monitor, or electrocardiography (ECG)-derived heart rate monitor on their body. On the other hand, emotion recognition

Sensor	Techniques	Emotions
Cameras	Facial Expression	6 Basic
		Neutral
Microphone	Emotional Speech	6 Basic
		Neutral
MEMS	Hand movements	Happiness
		Anger
		Neutral
Location	-	Happiness
		Neutral
Typing	Frequency	Anger
		Neutral

Table 2.3: Matching between smartphone sensor and emotion recognition technique.

using facial expressions or speech has limitations on its usage, because the device needs to be positioned in specific way.

As well as physical activity, there is not a concrete sensor to accomplish user emotional recognition. However, emotion recognition follows the same steps, every selected sensor ($s_1, s_2 tos_n$) may provide different sensor information ($rd1_1, rd1_2 tord1_n$).

Due to the high number of emotion context recognition approaches available, a possible way to deal with it is combining user information from more than one source. Hence, such information may be complemented easily using other techniques described above.

2.1.1.4 User Relationships

Inferring group activities is highly impossible without determine user relationships. First of all, all the user social relations may be included in a list called $friends_{SNS}$ and subsequently is check whose are closed. Not all user friends are important at any time or place. Hence, determining a Community of Interest (COI) (Rahman et al., 2009) based on the relation with them must be faced as a crucial task.

Note that COI is considered as an overlay network that extends the traditional social networks by assuming that a user's social network is people who are able to perform and action together. As users have different level of trust with their friends, we propose to implement a hyper-graph where each edge represents a relationship, such as close friend, family member, co-worker or other. Edges are briefly described below:

- COI_{family}: This list represents those people who belongs to your family. This list is created thanks to the SNS functionality, which allows to determine your siblings.

- COI_{work}: As well as family list, it is possible to retrieve those friend who work with you.
- COI_{closed}: This list contains people with whom you have more relationship in social networks. COI_{closed} may be contained people from COI_{family} and COI_{work} lists.
- COI_{other}: If someone do not belong to any list is included in COI_{other} list.

The main goal to split users in four COI's is to determine who are closest to the user and Therefore they are more likely to perform an action together. COI's are automatically updated, since social relations vary over the time, each list is modified according to the new interactions.

2.1.1.5 User Location

In the literature location is considered the most important kind of context above the others. However, as we previously said, location is more than a point in the space. Location information may have important information associated like work place, family homes, etc. Besides, orientation and elevation are also consider important location information in context aware systems. Knowing where the user is, allows applications to provide smarter and more precise information to the user. Using location information is possible to deduce spatial relationships between different users.

Determining which kind of location sensor use and trust is a matter of trade-offs in accuracy, speed, and of course energy-efficiency. Although GPS is the most accurate one and gives more location information from the user (Speed, bearing, altitude and so on), it presents some drawbacks in terms of energy consumption and availability. In context aware systems availability is an important point to take into account. In this case, GPS provide location information outdoors which reduce drastically its usage.

Moreover, GPS quickly consumes device battery energy due to the amount of information to be processed. On the other hand, using other Location Provider like Cell-id or even Wi-Fi, may work indoors and outdoors, responds faster, and uses less battery power, however the overall accuracy is lower. Hence, the selection of location sensor may be a crucial decision in context aware systems.

Like User Physical Actions module, location is normally collected synchronously, saving time that the location sensors are an active task. Since we do not know what kind of location system may collect information, depending if the user is indoors or outdoors, location information is stored in a duple with network identifier and the specific location value ($pos_{[gps,network]}$). Thus, location information is propagated to next level in a common and understandable way.

The main characteristics that have been used to create the location features vector should be: latitude, longitude, bearing, altitude, sensor (GPS or network) and text user location. Text user location is just a keyword, which describes the place where the user is, for example, it could be my home, at work, parents house, etc. This keyword should be demanded to the user every time the device is connected to a new Wi-Fi access point or stay a long period of time in a place (GPS information).

	Actions	**Emotions**	**Position**	**Social**
Raw Data	s_1rd1_1, $rd1_2tord1_n$ a_2rd1_1, $rd1_2tord1_n$	s_1rd1_1, $rd1_2tord1_n$	$pos_{[gps,network]}$	$friends_{SNS}$
Features	sma_{a_x,a_y,a_z} $mean_{a_x}$ $mean_{a_x}$ $mean_{a_y}$ $mean_{a_z}$ std_{a_x} $std_{a_y}std_{a_z}$ $corre_{a_x,a_y}$ $corre_{a_x,a_z}$ $corre_{a_y,a_z}$ $mean_{gpsSpeed}$	$ngram_{1,2}$	pos_{vector}	COI_{closed} COI_{family} COI_{work} COI_{other}

Table 2.4: Level 0: Raw Data and Features information

In short, level 0 produces a large amount of data which all of them is not necessarily to be transferred over network and shared in real time. That is why we could consider most of them in raw meaningful informational value. Also, processing sensory values with maximal refresh data rate would be unreasonable in terms of battery management. The main key to sensory networks is a balanced design of outlined high-level architecture.

We will presume that all the sensors are possible providers of user data. Fixing user information sources makes obsolete a general purpose context architecture in a short period of time. Hence, describing a general architecture able to collect information from every single source is a necessary task to accomplish. However, this work will be focus on the ones which are outlined in Chapter 1.

2.1.2 Level 1: Simple Context Actions

Level 1 aims to determine specific actions performed by the user instead of features or raw data given by the last level. The Simple Context Action is reported by implementing machine learning techniques (Decision trees, Naive Bayes, etc.) using raw data and features from level 0. In this level, user information is becoming less symbolic (raw data and features) to more specific information like running, walking, happy, at work, etc.

Therefore, Level 1 may be considered as a Activity recognition level which fetches the selected features in level 0 and classifies them in order to provide more specific information

from the user. Finally, the output of this level will be the current Simple Action in terms of Physical, Emotional, Social and Location context.

2.1.2.1 User Identification

At this point, identification process is almost finished since all data refers to a single user. However, to match this information with an explicit user in our system is a mandatory task to accomplish. Hence, level 1 output sum up to precedent identification information a user profile in our framework.

2.1.2.2 User Physical Actions

User context recognition using embedded sensors has enabled many context-aware applications in different areas, such as healthcare. Initially, one or more dedicated wearable sensors were used for such applications. However, recently, many researchers started using mobile phones for this purpose, since these ubiquitous devices are equipped with various sensors, ranging from accelerometers to magnetic field sensors.

An activity recognition module uses selected features to infer what activity is the user engaged in (Running, Walking, Standing or Motorized). Many different classification models have been applied to the problem of activity detection. There is no universally accepted method of detecting a particular range of activities and all techniques have associated advantages and drawbacks. Common methods include data driven approaches such as Decision Trees (DT), k-nearest Neighbour, Neural Networks (NN), Naive Bayes (NB) and Support Vector Machine (SVM) (Goldwasser et al., 1989). Some of those techniques were described on the Chapter 1, however, we will briefly describe some of them again.

2.1.2.3 Activity Recognition: Classifiers

Many different classification models have been applied to face the problem of people activity detection. This is highlighted by the studies included in Table 2.2. There is no universally accepted method of detecting a particular range of activities and all techniques have associated benefits and limitations (Wu et al., 2008b). Common methods include data driven approaches such as Decision Trees (DT), k-nearest Neighbour (k-NN), Neural Networks (NN), Naïve Bayes (NB) and Support Vector Machine (SVM).

- *Decision Trees*: (DT) are decision support tools using a tree-like model of decisions and their outcomes, and costs. The most implemented DT is the tree C4.5. Such systems

take as input a collection of instances or cases, each one belongs to one of a small number of classes and described by its values for a fixed set of attributes. Finally, the classifier can accurately predict the class to which a new case belongs. A basic decision tree approach to discriminate between four different activities using a smartphone is presented in (Miluzzo et al., 2008).

- *Nearest Neighbour (NN)*: The k-means algorithm is a simple iterative method to divide a given dataset into a specific number of clusters, k. The user decides the number k. k-NN algorithm is normally used for classification of activities based on the closest training examples in the feature space. Maurer et al. (Maurer et al., 2006) propose a multi-sensor activity recognition platform using MEMS on different body positions.

- *Naïve Bayes* (NB) is a simple probabilistic classifier based on Bayes' theorem. Given a set of objects, each one belongs to a known class, and each of which has a known vector of variables, NB aims to construct a rule, which allow assigning future objects to a class, given only the vectors of variables describing the future objects. In (Wu et al., 2008a), Wu et al. present a health care application when a patient is monitoring with various sensors located on their knee.

- *Support Vector Machines*: (SVMs) are supervised learning methods used for classification. Nowadays, SVM are considered a must try—it offers one of the most robust and accurate methods among all well-known algorithms. It has a sound theoretical foundation, requires only a dozen examples for training, and is insensitive to the number of dimensions. In addition, efficient methods for training SVM are also being developed at a fast pace. In the He et al. work (He et al., 2008) is shown how to accurate are three 3 different feature extraction methods using a SVM classifier.

Considering computational power and energy consumption restrictions, reducing the number of uploads should be mandatory. In that case, the tracking device preserves energy. However, selecting a the best technique is not easy but it is highly recommended in order to balance the energy consumption and the global precision of the system.

2.1.2.4 User Emotional Actions

As well as an user physical actions module, determining the techniques and features depends on the available sensors. It is not the same infer emotion from text that develop a speech recognition module to classify emotions with the voice. Nevertheless, it is mandatory to determine inferring techniques depending on the raw data collected in the previous level

Emotion Recognition Engine including Bayesian Network classifier categorizes incoming data into 7 types of emotions according to Ekman six basis and the neutral one: $emotion_{happiness}$, $emotion_{sadness}$, $emotion_{surprise}$, $emotion_{fear}$, $emotion_{anger}$, $emotion_{disgust}$ or $emotion_{Neutral}$.

To determine the current emotion of users, we adopted a machine learning approach consisting of the data collection, data analysis, and classification process. For these tasks, we used a popular machine learning software toolkit named Weka. The basic idea behind this level is selecting features that are "meaningful" to emotions instead of simply choosing words with high co-occurrence degree.

2.1.2.5 User Relationships

Some systems assume that COI members are determined a priory and only registered members can be invoked. However, automatically selection is possible, allowing the system find the most suited COI members. In our case, we decided to select those COI members according to the last context update. If our goal is to decide which users are performing a group action, we deleted those COI members which last context update was previous of the last 5 minutes.

Belong to an individual COI means that is possible that both users are doing an action together. Hence, each COI member may be active (perform a simple action) in the last 5 minutes and also is placed near to the user. We assume that the time limit is 5 minutes because of the selected gap between two different actions is this concrete value. Besides, in order to define if a user is near to another one is depending on the location measurement and the estimated error.

Select those active friends significantly reduce the COI vector size. Considering the average number of friends on SNS is over 100 and applying the temporary selection in order to determine those active users. At this point, COI members (COI_{active}) are considerably reduced. Moreover, thanks to the COI classification made previously (COI_{closed} COI_{family} COI_{work} and COI_{other}) the final set of friends give us much more information about the user.

2.1.2.6 User Location

Although, at this point user location information is really complete, however, adding new information according to the previous features is possible and the level 1 is the place where this information is inferred. As we previously described, determine if the user is indoors or outdoors may be a fundamental context information about the individual. The most effective method to

determine when the user is outdoors is by sampling the GPS signal and analysing the presence and quality of location information.

Because the app is developing under Android OS restriction, there are only two unique ways to determine if the individual is indoors or outdoors. The first one, is when the device is not able to connect with any satellite and the second one according to Android OS API is when the measurement error is over 50 meters. Therefore, the $pos_{[vector]}$ is transformed to $pos_{[information]}$ vector adding new relevant information such as indoors or outdoors, at work, at home, etc.

	Actions	**Emotions**	**Position**	**Social**
SPA	$action_{Running}$	$emotion_{happiness}$	$pos_{information}$	COI_{active}
	$action_{Walking}$	$emotion_{sadness}$		
	$action_{Standing}$	$emotion_{surprise}$		
	$action_{Motorized}$	$emotion_{fear}$		
		$emotion_{anger}$		
		$emotion_{disgust}$		
		$emotion_{Neutral}$		

Table 2.5: Level 1 output: Simple Actions

2.1.3 Level 2: Situation Assessment

Level 2 propose tries to find a contextual description of the relationship between objects and observed events in the level 1. According to the current Simple Actions (Level 1 output), it is considered which user information is relevant or not to the current user context information. Level 2 proposes to find a contextual description of the relationship between objects and observed events in the level 1.

This level involves deriving relations among user context information extracted in below levels, e.g., the user activity state (i.e. classification and location). Hence, this level may be faced as a data association problem, joining those user characteristics, which are relevant in that moment an place.

Data association uncertainty occurs when remote sensing devices yield measurements whose location and timestamps are not the same, that is, not necessarily the SCA of interest. This uncertainty occurs when the context providers report user context information from different places or time. Hence, in order to detect it, the available SCA has to be selected, in which case each SCA are selected by proximity in time (Temporal aggregation) and place (Spatial aggregation).

Consider the problem of associating each SCA from a individual ($action_x$, $emotion_y$, $pos_{information}$ and COI_{active}). Every user SCA provides their local state (timestamps, place and

user ID). We are interested in grouping whether these N SCA are relevant user information at this point. This procedure avoids searching for those SCA's from the user in the entire SMA space. However, a SMA in the gate, while not guaranteed to have relevant user context information associated with the gate (position or time), is a valid association candidate, and such a gate is called a *validation region*.

In the spatial aggregation we track all actions falling within a circle of given radius (called Coverage Circle Radius, CCR) centered on each landmark point. Data values are weighted by their distance from the landmark point (i.e. from the center of the circle) using an inverse exponential function and annotated with their timestamp. The validation region is set up to guarantee that the user SCA falls in it with high probability, called the *gate probability*, based on the statistical characterization of the predicted SCA. Each SCA outside the validation region can be ignored because they are too far from the predicted another SCA and thus unlikely to have consistent user context information. This situation occurs when the gate probability is close to unity.

Temporal aggregation is achieved by aggregating all roughness SCA computed for the same position on the gate or CCR. Values contributing to the same point are sorted by descending timestamp. The contribution of each roughness value decreases exponentially in time, thus the latest computed value has the highest weight, while older values are steeply down-weighted. This exponential decay is simply implemented by updating the temporal estimate (minutes temporal gate is a reasonable time horizon in grouping actions) as the average between the current value and the previous aggregated estimate.

All the SCAs taken by the user would infer a global action with any relation between the other ones. For example, raw data and features promotes running, happy, location information and individual COI for a particular user. Level 1 output promotes relevant user information kike Running, location and one COI active member. Maybe all these SCA do not make sense in an individual way but all together may infer that the user is doing exercise with someone (Figure 2.2). Therefore, the final output of the inContexto level 2 is the user context information which is relevant in terms of time and location association.

2.1.4 Level 3: Action Context

Action Context activities (high-level activities or group activities) are composed of a set of Simple Context Actions, e.g. shopping can consist of driving car, walking, standing, etc, also, in a meeting can consist of two people talking together. Moreover, Action Context usually last longer than low-level activities, they can last up to a few hours. Furthermore, with the

knowledge obtained, the recognition of his Simple Context Actions can be improved. For example, if the recognized high-level activity is shopping, during this time the probability of the activity walking is much higher than Nordic walking.

The explicit goal of the last level is to enable the recognition of longer-term and high-level activities. According to the current user information ($action_x$, $emotion_y$, $pos_{information}$ and COI_{active}) this level generates predictions thanks to the implemented reasoning model. Beyond the standard reasoning model based on the ontology mechanism, it is possible to perform rule based inferences using a description logic inference engine.

In our first approach, the selected inference engine is based on ECA (Event-Condition-Action) paradigm which is composed by a defined set of reactive rules working over an event-driven architecture (Michelson, 2006). User context (event) triggers actions depending on the given condition (ECA rules). An ECA rules is divided in three different parts: the*event* is the signal that triggers a set of rules; the *condition* which if is satisfied makes the execution of the rule to continue and finally, the *action* defines the execution flow of a process. As we previously explained, three fields composes an ECA Rules (events, conditions, actions) and they are described below:

- *Events*: describes a situation (user activity or location in this case) to which the rule may be able to respond. Events can be essential divided into two categories: (i) primitive events, which correspond to elementary occurrences, and (ii) composite events that are composed for more than one primitive events.
- *Conditions*: specifies the conditions to trigger the ECA rule. Once the result of the condition evaluation is true, the condition is satisfied and the action field is executed.
- *Actions*: describes the task the rule considers relevant to the event and the condition. Actions field indicates the subsequent activities if the condition is satisfied.

Summarizing, a ECA rules is divided in *events* which are the signal that triggers a set of rules; the *conditions* which if is satisfied makes the execution of the rule to continue and finally, the *action* defines the execution flow of a process. At the beginning, are manually created in order to teach the system, however the system will be learning new rules an creating automatically. ECA rules engine is constantly evaluated in order to detect configured events, executing the associated actions if the conditions are fulfilled.

To specify the set of rules by an expert, which one triggers actions, and depends on the main goal of the application is greatly necessary. In (Gil et al., 2012d) is presented a set of

rules in a health care scenario where the high levels actions triggers alarms when a patient perform a forbidden action.

2.2 Smartphones and Social Network Sites As Sensors

Obtaining people context has been widely studied for decades, especially in recent years was made analysing automatically ongoing activities from video sources. Nevertheless, from a technological point of view process video is computationally costly. Besides, there is a new kind of personal tools which has been set up, and equipped with sophisticated tiny sensors and advanced computing hardware. All this advantages may be used to infer user's context: location, activity, emotions, social relations and so more.

In the last decade mobile phones have reached every part of the world, and 86% of the world's population had a cellular subscription in year 2012 (Van Dijk, 2012). Nowadays, phones serve for travel planning, staying in touch with friends using social network sites, on-line shopping and numerous other purposes. Most of the services available on smartphones require an Internet connection (e.g. social networking and email), creating communication opportunities for sending collected data.

The approach based on personal mobile devices maximizes the amount of collected data with no extra costs, need for maintenance of the devices neither in a obtrusive way. For these reasons, the focus of research has been driven up to mobile sensing, using and exploiting these devices capabilities.

In contrast to conventional recognition context systems where raw input data is generated by calibrated electronic sensor (inertial or video cameras) with well-defined features, soft sensors are presented also as perfect tool to retrieve user context information. Soft sensor considers combining human-based data expressed preferably in natural language, which give researchers an opportunity due to the incredible popularity of Social Networks Sites (SNS).

Combining soft and hard user data is even more challenging yet since the information from different sources may be contradictory. However, the new trend towards a more general where both human and non-human sensory data can be processed efficiently and inContexto presents a new way to afford this problem.

2.2.1 Hard sensors: Smartphones

Smartphones have become ubiquitous personal devices since most of them are equipped with a variety of built-in sensors for human-computer interaction, the user's keystroke behaviours

can trigger these sensors to gather the data of behavioural characteristics without extra hardware devices. Normally, smartphones are composed of a screen, a microphone, cameras, an accelerometer, GPS receiver, a digital compass, a gyroscope and a communication module.

Thanks to these sensors developers are expanding the boundaries of mobile applications. Despite the recent phenomenal progress, the area of mobile personal devices promises further advances as sensing and processing capabilities of mobile phones grow.

Thanks to rapid technological developments, integrating numerous sensors and communication interfaces, these devices can be exploited for data collecting in urban environment and about population motion. In addition, the high and growing number of users of these devices maximizes the number of observed individuals, and may reach millions of devices distributed throughout the world. The participation of people becomes simpler, with no need of carrying specific equipment since participants already use their personal devices daily.

New applications for smartphone rely on in-built sensors. For example using the microphone is possible to commands for operation, search for information on Internet or even detect if someone is talking around it. Another good example is the tri-axis accelerometer, in this case detecting the motion is possible to count user steps (as a pedometer) or detect smartphone position in order to change the screen orientation. Moreover, the gyroscope and digital compass may be integrated to help navigation when GPS signals are weak or not present.

However, with the advent of smart mobile devices that possess rich sensors, continuous connectivity and push notifications, it is now possible to continuously share information with other devices, in real time. Hence, it is possible to assert that the smartphones feels the same forces, travels at the same velocity, is about the same temperature, is exposed to the same sounds that the user who is carrying it. Although new kind sensors are continuously introduced in the top-level devices, this section will focus on those sensors that are being used in services and platforms that ultimately implement contextual services.

1. Accelerometer: A tri-axial accelerometer is a sensor that returns a real valued estimate of acceleration along the x, y and z axis. Usually, accelerometer origin of coordinates is placed in the lower-left corner with respect to the screen, with the X-axis horizontal and pointing right, the Y axis vertical and pointing up and the Z axis pointing outside the front face of the screen.

   This representation and mobile phone position present some drawbacks in order to detect properly user movements. An example of this problem is when the smartphone is worn on a pocket. It is not clear which axis or axes is representing the gravitation, to solve this problem is necessary to transform accelerometer data, using digital compass.

Accelerometers are suitable as motion detectors as well as for body-position and posture sensing (DeVaul and Dunn, 2001) since acceleration data of walking or running displays distinct phases and periodicity of the signal however it is very difficult to differentiate transportation modes.

2. Digital Compass provides two kind of measures: the first one is the orientation which its values are in radians/second and measure the rate of rotation around the X(roll), Y (pitch) and Z (yaw or Azimuth). Also, the coordinate system is the same as is used for the acceleration sensor. Digital Compass reports the angle between the magnetic north and the mobile phone's Y axis (orientation measurement). All values are in micro-Tesla (uT) and it measures the ambient magnetic field in the X, Y and Z-axis. This sensor does not have a concrete value describing user actions, but it is usually used to determine user movements' direction.

3. Gyroscopes are the most commonly used sensors for measuring angular velocity and angular rotation in many navigation and homing applications. They measure how quickly an object rotates, specifically; measure the rate of rotation around the X, Y and Z axis. The coordinate system is exactly the same as is used for the acceleration sensor. Gyroscopes are the only inertial sensors that provide measurement of rotations without being affected by external forces, including magnetic or gravitational or fabrication imperfections.

4. Location sensor: There are three ways to locate the smartphone, first of all using a GPS, in this case every smartphone provides an Assisted GPS (van Diggelen, 2009). A-GPS improves the performance by adding information, through another data connection (Internet or other) than unassisted GPS. In order to receive and process signals is computational costly, minimizing the amount of time and information required from the satellites. The A-GPS receiver uses satellite to locate itself but it can do more quickly and using weaker signals than an unassisted GPS. Normally, an A-GPS provides 2-4 meters error.

   The second way to locate the smartphone is using GSM cell tower triangulation. This technique reduced as well as accurate than GPS however, the energy consumption is reduced as well. According to the application goals, to balance the accuracy and the energy consumption is necessary. Besides, a coarse location (GSM) could be enough instead of a precision location (GPS).

   Finally, Using Internet connection (Wi-Fi) is possible to locate the smartphone thanks to W3C has reloaded a Geolocation API to standardize an interface to get back the

geographical location information for a client device [1] .

5. Networks Sensors: A number of communication options are available for transferring the results to the backend server of a typical mobile phone device (e.g., Bluetooth, HTTP+3G, HTTP+WiFi, see table 2.6). Connectivity is a basic factor, which influences the necessity of transmitted data. Due to its multiple communication technologies these devices are able to send the collected data in a simple way, resulting in lower costs compared to other devices.

6. External Sensors: For now, smartphones are easily programmable devices with multiple functionalities by operating the application software to control the built-in sensors. Despite the prominent properties of smartphones, there still existed a lot of problems to restrict the sensing of environment, physiology and health-care.

   Fortunately, some external sensors are developed to provide the detection tasks and connect with smartphones using communication sensors. For example, new sensors that could be connected to smartphone are: air pollution, air temperature, humidity, air pressure, chemicals, physiology signal, and cardiovascular. The main goal is not only extending the usability of smartphones but the sensing capability anywhere and anytime. Based on the requirement, the approaches of how smartphones can be equipped with more features as an instrumentation tool for measurement and monitoring will be introduced in details.

Technology	Bandwidth (D/U Mbps)	Latency (ms)	Download (kB/s)	Upload (kB/s)
EDGE(2,5G)	1.3/0.6	100-1,200	3-10	2-5
UTMS(3G)	28/11	50-500	20-50	10-40
HSDPA(3,5G)	42/12	-	-	-
LTE(4G)	100/50	-	-	-
WiFi	10/54	10-100	400-1000	300-900
WiMax	46/4	-	-	-
BlueTooth	2/2	-	-	-

Table 2.6: Communication sensor in mobile devices.

2.2.2 Soft sensors: Social Network Sites

The proliferation of sharing on social networks, such as Facebook, Pinterest, Linkedin, and Foursquare, has closely connected people more than ever before. Social Networks Sites (SNS)

[1] Geolocation API http://dev.w3.org/geo/api/spec-source.html

are far from mere people social tools they were in the past. Daily, Every day hundreds of millions of people log onto social network sites and share their latest updates creating online repositories of accessible user information. Social Networks Sites Sites (SNS) are increasingly popular these days. In (Boyd and Ellison, 2008) is described Social Network Site as:

web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system. The nature and nomenclature of these connections may vary from site to site.

Each SNS is implemented with specific features, however all of them have a common point, which consist of visible profiles. Daily, SNS users share their personal information; SNS manage uncountable gigabytes of useless user information. Why do not we use these data to obtain user context information?

Typically, User profiles include descriptors such as age, location, interest and so on. User profiles are becoming more precise: Music preferences, movies, clothes, friendship relationships, personal agenda, etc. In (Ellison et al., 2007) is described social network site as a Web-based services that allow individuals to:

1. Construct a public or semi-public profile within a bounded system.
2. Articulate a list of other users with whom they share a connection.
3. View and traverse their list of connections and those made by others within the system.

The nature and nomenclature of these connections may vary from site to site. However, all of them have a common point: User Profiles. User profiles include descriptors such as age, location, and interest schools attended. User profiles are becoming more precise: music preferences, movies, clothes, friendship relationships, personal agenda, and so forth.

2.3 Smartphone Context: Entity concept in a Mobile Scenario

In this section, we outline some of the challenges involved in organizing and conducting the collection of user context data whether it is from hard and soft sensors. The user's surrounding information is also known as context. Context, in general, can be used to overcome the problem of systems retrieving too much information that may not be relevant in the user's current situation and also to minimize their involvement in interaction particularly on a mobile device.

The principal problem is to decide those features, which define precisely user context, specially, in a mobile scenario where user information is generated rapidly. Before proceeding to define the entity concept, it will be necessary to properly describe context. As was pointed out in the State of Art Chapter 1, we believe that Dey definition of context is most precise describing a Mobile scenario. The term Context is defined by Dey et Al.(Dey and Abowd, 2000) follows:

Any information that can be used to characterize the situation of entities (i.e., whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves.

Dey context definition introduce a new term "entity", an entity is the minimal concept. Regarding to Dey definition of context, an entity can be represented by anything that is important to the context aware application. Therefore, in terms of people context, everything in the world may be considered an entity. However, since we are dealing with mobile environments, in this dissertation we will focus on those objects with provide Internet connection and sensors to assess their condition or the environment itself.

Moreover, each entity is characterized by four categories (Identity, location, status or activity and time) According to Dey's definition about context and entity, we may redefine the way to interpret people context using smartphones. (See in figure 2.4).

In 2.4 is depicted an overview of the person entity taking into account the mobile scenario and wearable sensors. Entity concept is the result of combining all the user context information: Identity, Location, Activity/Status and Time. By studying people data from smartphone sensors and user information on SNS is now possible to gather new and most precise user information. In supporting these features, context-aware applications can utilize numerous different kinds of information sources.

Once we have classified context information, there is a need of sensing it for being able to be used by systems and applications. The context information needed may be obtained from a variety of sources, such as networks, devices, or by applying sensors and browsing user profiles. Now, we are going to discuss which is the best way to collect this information using smartphones and social network sites.

2.3.1 Identity Context

Although it seems obvious to comment that identification in a context application is crucial, not only is important to determine whom is doing an activity. A good identification opens a wide

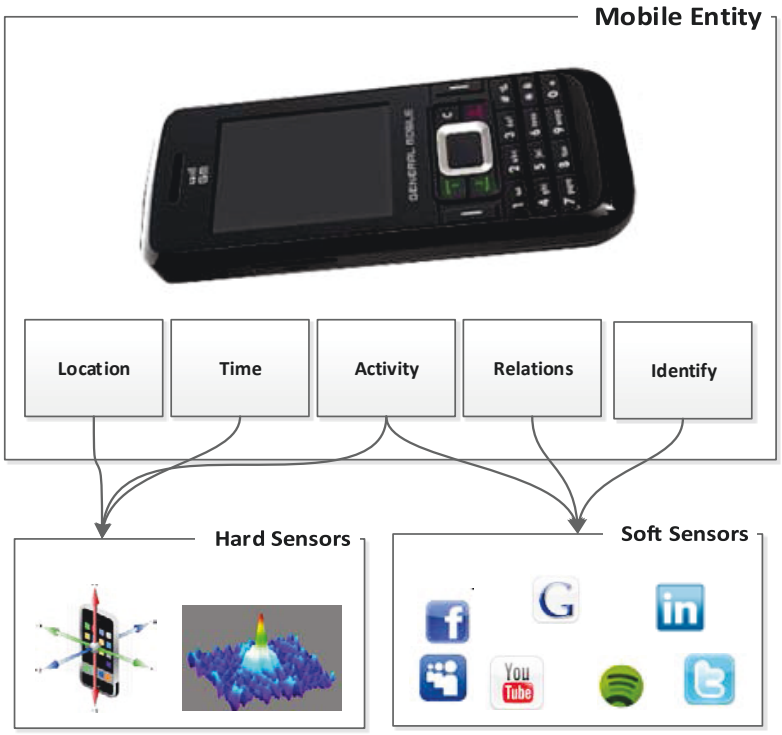

Figure 2.4: Entity representation using smartphones.

range of new possibilities in order to offer new context based services or improve existing ones. For example, daily users perform the same activities, if the user identification is performing correctly, the system can store these actions, creating a user activity log. Afterwards, it may anticipate user movement offering more adapted information. Hence, each user stored activity or status can be associated with another activity that the user may engage in.

The identification process may be implemented basically in two ways: via software or hardware. Hardware identification may be performed using the terminal identification such as MAC address, telephone number, etc. Normally, such identification is very common in stationary or external devices, which are characterized by their durability over time and its low turnover rate. Inside this group of devices can point out as most important: Zigbee devices, meteorological stations, Wi-Fi access points, etc. However, this method presents some drawbacks because of smartphone itself does not identify the concrete person who carries it. Hence, if another user manipulates the same device, there will be identification problems.

On the other hand software identification allows to identify uniquely the user who is on

the scene. Software identification methods currently available are basically focus on two types: User credentials (user-name and password) and the delegation of the identification process to other applications, usually social networks sites (Kaila, 2008). Facebook, Google, Twitter to name a few provides a system (OAuth or OpenID) that allows to any application identifying every user in theirs Social Network Site.

In this way, to delegate the identification process to SNS being much safer than creating a new identification process. Besides, Users trusts in SNS it is a system that the user trusts. From the end user viewpoint these systems are almost transparent. The user will just be redirected to the identification system site and then redirected back after having successfully authenticated.

2.3.2 Location Context

Location aware could be the main factor in the development of context applications. Nevertheless, physical objects and devices are spatially arranged and humans move in mobile and ubiquitous computing environments. Location-aware is only one aspect of context aware as a whole (Schmidt et al., 1999b). Location context may be described as an application dependent on the geographical location. Location answers the question where is the action or status taking place? As well as happened with the entity identification process, entity location not only answers the previous question, there are so many issues that have a good location system may help to retrieve better contextual user information.

Locating using mobile phones is really simple in outdoor environments where GPS provides a good solution to determine the location of mobile devices however in GPS-denied areas such as urban, indoor, and subterranean environments, unfortunately, an effective solution are not available yet. However, GSM, Wi-Fi, and Bluetooth networks are even used for people localization indoors.

2.3.3 Status or Activity Context

First of all to define a difference between user status and smartphone status is extremely necessary. Smartphone status mainly refers to communication behaviour: Calls and calls attempts, sent and received SMS, SMS content, battery level, wireless connections, etc. Although, mobile device activity may be important is not as important as the user activity. On the contrary, user status or activity does not refer just to human physical activity; even it refers to user emotional state. User moods may be extremely necessary in order to offer good services (Cowie et al., 2001) since daily people makes decisions regarding their emotional state.

The activity context covers those activities that the entity is currently and in future involved in. Besides, it answers the question What does the entity want to achieve and how? It can be described by means of explicit goals, tasks, and actions. The context generation process using raw data mainly consists on the following phases. Raw sensor data, such as those from phone's accelerometer, are seldom of direct interest without transform in more descriptive information.

Machine learning techniques are usually employed to infer higher level concepts, for example, a user's physical activity (Tapia et al., 2007) or emotions (Miller, 2012). For the inference to be made, firstly, we need to decide which aspects of the raw sensor data are the most representative of the phenomena we would like to explain. Then, appropriate machine learning techniques are develop to build a trustworthy model of people behaviour and train it with the data collected. Finally, the current user context is inferred from the collected data.

2.3.4 Relations Context

The relations context describes the relations between two different entities. As well as Dey's definition of entity, such surrounding entities can be persons, things, devices, services, or even information. The set of all relations of the entity builds a structure that is part of this entity's context. A relation expresses a semantic dependency that the entity has established to other entities and it shows a certain circumstances that these two entities are involved in. Potentially, an entity can establish any number of different relations to the same entity.

A relation could be consider when the contexts of two entities overlap and parts of the context information become similar and shared Potentially, an entity can establish any number of different relations to the same entity. Additionally, relations are not necessarily static and may emerge and disappear dynamically.

According to the Zimmerman et al. (Zimmermann et al., 2007) definition of context a relation could be subdivided into three different relations (social, functional and compositional):

- Social Relations: It describes the social aspects of the current entity context. Usually, interpersonal relations are social associations, connections, or affiliations between two or more people. For instance, social relations can contain information about friends, neutrals, enemies, neighbours, co-workers, and relatives. In our case, this relation category is the most relevant according our work.

- Functional Relations: A functional relation between two entities indicates that one entity makes use of the other entity for a certain purpose. For example, such relations

exhibit physical properties like using a hammer, sitting on a chair or operating a desktop computer.

- Compositional Relations: It is the entity relation with a whole and theirs parts. In the aggregation, the parts will not exist anymore if the containing object is destroyed. For example, the human body owns arms, legs, etc.

2.3.5 Time Context

As well as location activities taken by the user, time context does not have any meaning if it is not possible to determine the action in time. For that reason time is also an essential characteristic in context aware applications. Not only time is a essential aspect in context aware applications, for the human understanding and classification of context, statements are related over the temporal dimension (Gross and Specht, 2001). The same action could have a different meaning depending on the time of the day or even in a different week day. For example, driving a weekday in the morning probably means that the user is going to work, on the contrary in the afternoon means that he is going back home.

This category subsumes time information like the time zone of the client, the current time or any virtual time. All smartphone developer API provides a way to access to the standardized representation of time CET (Central European Time). Overlay models for the time dimension are often applied in context-aware computing and provide categorical scales like week, working hours or weekends.

2.4 Summary and Conclusion

The most critical part of this work has focused on developing a framework to infer and evolve people context information from multiple sources. inContexto is a layered architecture (four levels) where user context information evolve from raw data (Level 0) to more specific information or group actions (Level 3). Bottom levels are just a collecting module (raw data and signals) and every single level the information flows the user context information is richer and clear.

Moreover, a mobile device entity is defined according to Dey's and Zimmerman's definition of context. An entity is defined as a mobile device, which provides hard or/and soft sensors, provides Internet connection everywhere and is portable. Activity recognition systems identify and record in real-time selected features related on user activity using a mobile device.

Considered future works extending the development of every single module also it will extend level subscriptions methods in order to generate information autonomously, for example, a new level 1 activity classifier is develop, defining the needed features and subscribing a user to this new classifier the system may automatically force to use it if the previous level provides the classifier features.

Another important line of research to consider is to use top-level output as a raw information in the first level. In that way is possible to enhance the overall precision. Next chapters will describe the algorithms implemented in level 1 in order to perform activity and emotional recognition and also it will be provided an inContexto example.

3

Inferring user activity and emotion context

THIS chapter focuses on Level 0 and Level 1 of the activity recognition module and it is divided in two parts: The first one corresponds to an evaluation of the selection the raw data, features and algorithms used by inContexto in the first two levels. This algorithm was developed taking into account the impact of mobile phone limitations (Energy consumption a processing) and providing a way to minimize them. Considering that the creation of a adequately dataset is extremely difficult, the generated dataset was created with real information acquired from 8 people and subsequently, with all this data simulated trajectories are generated in order to complete the dataset.

The second part is organized in the following way: firstly, section 3.1.1 presents how raw sensory data is collected, describing the subsequent processing steps in detail. Secondly, being the main focus of this part, is the feature selection step, where different features are compared and also new algorithms are introduced. Section 3.1.2 defines the selected features and the section 3.1.5 describes the developed algorithm used to measure the performance and to evaluate the presented techniques. Finally, section 3.1.6 shows the final results of the classification process technique of each created dataset.

3.1 Study of Activity recognition using smartphone accelerometer

This section presents the data processing used in the physical activity recognition used in inContexto architecture depicted in the previous section (Section 2). This process can be described as a chain of processing steps, starting from collecting raw sensory data and resulting

in a prediction of an intensity and activity class. There are many different methods for retrieving activity information from raw sensor data in the literature (Guiry et al., 2014). However, in this thesis we follow the classical approach whose main steps can be categorized as: pre-processing, segmentation, feature extraction, dimensionality reduction and classification (Krishnan et al., 2009). Figure 3.1 summarizes the activity recognition process.

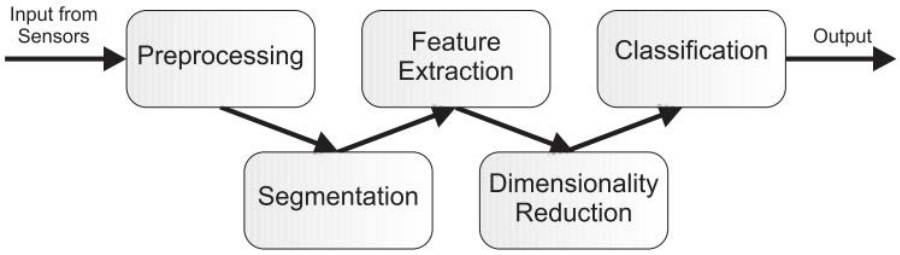

Figure 3.1: Hard sensor smartphone.

3.1.1 Making the Datasets

The first task that to solve in activity recognition systems is to find a good dataset to test the methods and techniques developed. In this case, user activity information were recorded when they were wearing the smartphone placed in their hip pockets. The collecting groud was formed by eight male and female participants between the age of 20–37 years. They have been participated as subjects for the empirical data collection experiments. Users were encouraged to wear the device as much as possible while they were performing three different activities: running, walking, and standing up (Stopping).

Once the data was collected. the labelled problem was faced. The ground truth labels were mainly added autonomously, thanks to the power of the GPS. The GPS speed value was used to tag every single action that user took. However, in order to preserve the dataset quality, data was solely recorded and tagged when the GPS achieved less than 10 meters of accuracy. Thus, from every participant action (running, standing, and walking), which has been taking place outdoors, the data acquisition, module recorded the speed, accuracy from the GPS (auto tagging) and also the selected features.

Typically, the best practice to acquire data preserving energy consumption is to select the lowest sampling frequency rate and switch to a higher one upon detection of an interesting event. In our case is necessary to continuously gather user information because everything is important. Therefore, the selected sampling frequency should be constant and according to (Henriksen et al., 2004) study, the sampling frequency range requiring to obtain human activity

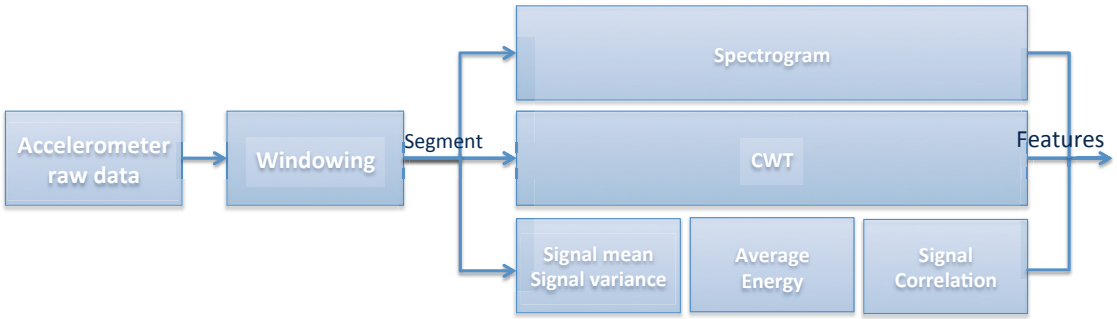

Figure 3.2: Dataset Generation Process.

context is $0.6Hz$ to $2.5Hz$. Besides, taking into account the aliasing problem and following the Nyquist-Shannon sampling theorem, the sampling frequency rate should be at least $5Hz$.

$$F_E \geq 2 * F_{Max} \tag{3.1}$$

Sampling frequency is not clear in Android OS since it provides only four different sampling frequencies (Fastest, Game, Normal and UI) and the value is not constant. The value depending on the computational workload of the smartphone but normally fastest sampling frequency is 50Hz. Besides, accelerometer and GPS raw data have been stored into a sliding window of 512 samples (Approximately 5 seconds), 256 of which overlap with consecutive ones. A sliding windows with 50% overlap has been defined in previous works (Bao and Intille, 2004). Extracting features from a window is a fairly effective way to preserve class separability and can represent the characteristics of different activity signals in each window.

In order to provide an effective and efficient description of patterns, pre-processing is often required improve performance, removing noise and redundancy in measurements. In this study, the accelerations and azimuths of the pedestrian were mainly collected with a smartphone with Android Operating System. Android OS provides four different sampling frequencies. This frequency are not fixed and depends on the Operating system, there is no control over it. The interfaces sampling can be performed continuously or periodically (Lee et al., 2014). In continuous mode the sampling tasks are executed again as soon as the previous task returned the result, while in periodic mode the execution of the sampling tasks are controlled, for example, by timers. Pre-processing module aims to simplify the operation in the feature selection module. Hence, raw accelerometer data is stored in a sliding window of 512 samples (Approximately 5

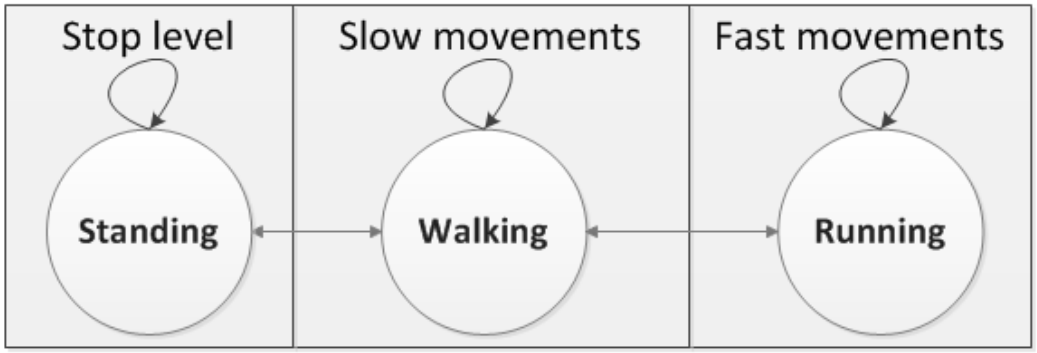

Figure 3.3: Trajectory generation flow chart.

seconds using $50Hz$ frequency sampling rate), 256 of which overlap with consecutive ones. An sliding windows with 50% overlap has been defined in previous works (Bao and Intille, 2004). Every featured vector contains different values from the raw accelerometer data: 3 axes mean values, 3 axes standard deviation values, correlation between each axis and signal energy for each axis.

	Running	**Standing**	**Walking**
Instances	150,718	345,318	240,825
Minutes	32.36	77.42	40.5

Table 3.1: Dataset duration (min) and samples for each activity.

Since it is very complicated to generate a meaningful dataset, normally, a big amount of samples or trajectories to start the training phase is necessary. This process is quite costly ant tedious for the participants; they have to repeat every day the same action in different situations. For that reason, in our case we propose to generate artificially more complex trajectories using the information collected previously (Show Table 3.1). The artificial trajectory generating process consist in 3 different steps as Image 3.3 shows. Firstly, the dataset is stored in 3 different files; each one corresponds to a specific activity (running, walking, and standing up). Subsequently, a Java program has been developing to mix every activity in an unique trajectory taking a bunch of information of each file. Normally, the number of samples is restricted between 2-7 seconds. Besides, no all combinations are allowed, for example if you are running, stop may not be the next action firstly should be walking. This restrictions are depicted in the trajectory generation flow chart (Image 3.3). Finally, all the generated trajectories, whose maximum length may be 70 seconds, will be stored in a new file in order to

start the pattern recognition process. Next, we will summarize some requirement to take into account to generate as much real trajectories as possible:

1. All the trajectories start with a standing up action;
2. the next action could be the action besides (Figure 3.3) or the same action again;
3. the minimum duration of each action is 2 seconds and the maximum is 7 seconds;
4. Finally, each trajectory consists in 10 actions (Maximum length 70 seconds, and minimum 20 seconds).

Finally, the generated dataset consist of 1000 different trajectories created using the collected data (Table 3.1). In terms of duration and actions performed every trajectory is completely different. Then, when the trajectories generation process is over, we discretized each trajectory according to the GPS speed value. Thereby, it should be possible to perform classification methods as J48 tree. Thus, all the samples are discretized in 5 classes as follows:

1. *Stop*: This class contains those trajectories whose GPS speed is less than 1 km/h.
2. *Walk*: In this case, GPS speed limit has to be more than 1 km/h and less than 4 km/h.
3. *Walk Fast*: The third class includes those trajectories whose GPS speed limit is among 4 and 6 km/h.
4. *Run*: A running trajectory is one whose GPS speed limit is more than 6 km/h and less than 10 km/h.
5. *Run Fast*: Finally, we consider running fast every time a trajectory with more than 10 km/h is performed.

Figure 3.4 shows raw acceleration (from the smartphone IMU, up-down direction) from one subject, while performing the various activities defined in the dataset's data collection protocol. These plots show that different tasks have a different signature; most of them can be easily identified by visual inspection. Therefore, it is clear that with the appropriate data processing steps (e.g. the right features extracted or the right classifiers chosen) these differences can be captured, making activity recognition and intensity estimation possible. For example, the posture standing can be easily distinguished from the postures walking and running with just the mean value of the presented acceleration signal. Another example is that activities including steps (walking-related activities) can be distinguished from other activities.

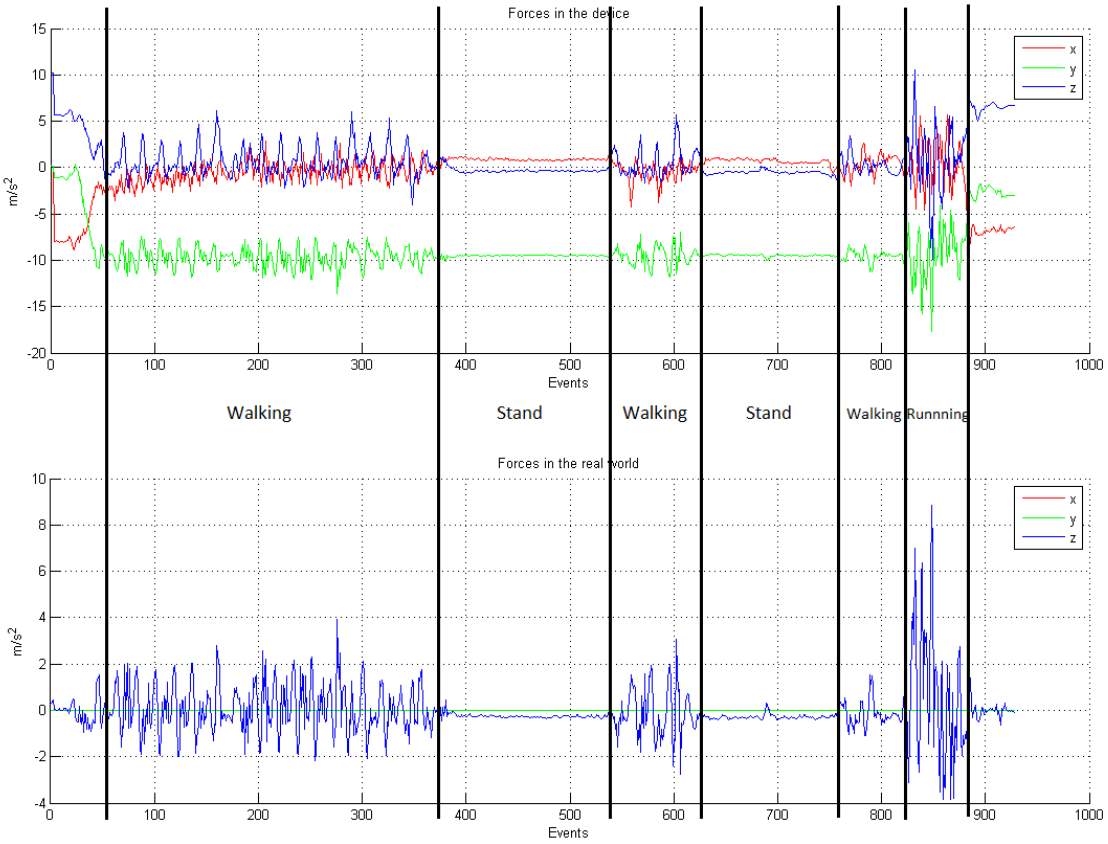

Figure 3.4: Example raw acceleration smartphone IMU from the dataset.

3.1.2 Selected Features

The goal of this chapter is to define which features are well suited and to identify basic or recommended activities and postures. These goals are motivated by various health topics as discussed in Chapter 1. For these purposes three different dataset have been created depending on the features selected (Short-Time Fourier Transform, Continuous Wavelet Transformation and Statistical features). These datasets will be used in this chapter to apply different data processing methods and classification algorithms, and to create a benchmark of physical activity classification problems.

As we described Chapter 2, in the literature, there are mainly two types of features from accelerometer raw data. The first one is related to frequency properties (DWT, CWT, and STFT). The second one is the technique that creates a vector with statistical features (SMA, signal mean, correlation, etc.). From the frequency features we selected two of them. The first one is based on Spectrogram function (STFT, Short-Time Fourier Transform) and the second dataset was created using information from the Continuous Wavelet Transformation (CWT).

In both cases (STFT and CWT) present several values, there are more than 150 frequencies for each technique. However, all of them are not necessary, we need just the more representatives ones. As we can see in the Figure 3.1, there are some frequencies without relevant information, even it could be consider repeated information. For that reason, our vector from frequency dataset contains the first 25 frequencies. The active frequencies (Red and yellow bars) depending on the action taken (Steady actions or not steady action) are clearly distinguishable. Non-active frequencies are coloured in blue, which represent when the user is doing a sedentary action (Standing, sitting, etc.).

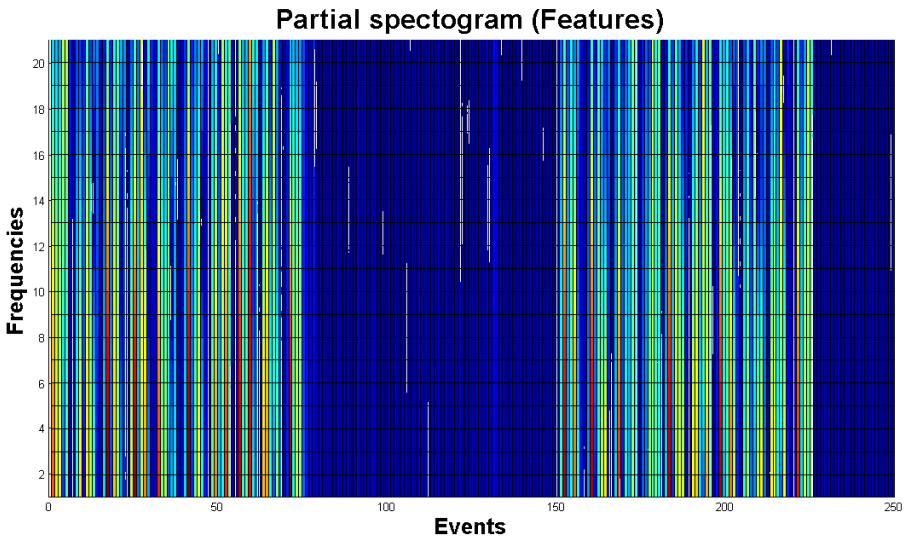

Figure 3.5: Partial spectrogram (Features selection).

Moreover, in order to compare the performance of every kind of features an statistical dataset was developed. In this case, the dataset consist of eight different features from the accelerometer values. These selected values are: signal mean, correlation between axes, energy and variance, which are usually, extracted from the triaxial acceleration data.

	CWT	**Spectrogram**	**Statistical**
Features	25	25	12

Table 3.2: Number of features for dataset.

3.1.3 Preprocessing

A low-level sensing module continuously gathers relevant information about the user activities using embedded sensors in the smartphone. The module just acquires data from the compass and the accelerometer sensors. As we previously comment, the accelerometer provides the forces (static and non-static) acting on the device. It returns a three component (x, y, z) vector that represent the three-axis forces acting on the device Cartesian reference system (See Figure 3.6). Note that the accelerometer reference system is also constantly changing due to device's position. Accelerometer and compass data is accessed through Android OS API, in concrete SensorManager class which provides methods to obtain all the mobile sensors. A low-level sensing module continuously gathers relevant information about the user activities using sensors. Thanks to Android OS provides background processing, it is possible to run services without human control.

Figure 3.6: Smartphone coordinates origin.

In frequency datasets should be unified the accelerometer values to a common origin system.

For that reason, 3-axis accelerometer value is transformed to a Real world acceleration values. Computing the inclination matrix I as well as the rotation matrix R transforms a vector from the device coordinate system to the world's coordinate system, which is defined as a direct orthonormal basis. I matrix is a simple rotation around the X-axis and the rotation matrix R which is the identity matrix when the device is aligned with the world's coordinate system a real world.

$$a_{realworld} = a_{smartphone} * R * I \tag{3.2}$$

Where I matrix is a simple rotation around the X-axis and the rotation matrix R which is the identity matrix when the device is aligned with the world's coordinate system.

Processing acceleration data may have some advantages, for example, in these cases, which the smartphone is regularly subjected to acceleration forces that correspond to one or more particular profiles, but where the mobile device may have a different orientation each time it is subjected to such forces. In particular, for example, a mobile device that is carried in a purse, handbag, briefcase or laptop bag; or a mobile device resting on the seat or console of an automobile. By translating the acceleration data, profiles may be advantageously matched and employed without requiring a user to maintain the mobile device in a particular position or orientation.

Figure 3.7 represents the device accelerations and shows the changes of the three forces depending on the movement taken it by the user (Running, Walking, Standing). On the other hand, Figure 3.8 represents the transformation from the smartphone reference to the real world reference. This work uses GPS in order to obtain the speed of the person who is taking place the action, thus, the classifier output value is the mean of the speed in the sliding window.

3.1.4 Segmentation

To obtain at least two or three periods of all different periodic movements, a window length of about 3 to 5 seconds is reasonable. For example, experiments presented by Lara et al. (Lara et al., 2012) showed best results with a window size of 5 seconds when using acceleration data for physical activity recognition. Therefore, and to assure effective discrete Fourier transform (DFT) computation for the frequency domain features, a window size of 512 samples was selected. Since the sampling rate of the raw sensory data is about 50Hz, the segmentation step results in signal windows of approximately of 5 seconds length. Therefore, the pre-processed

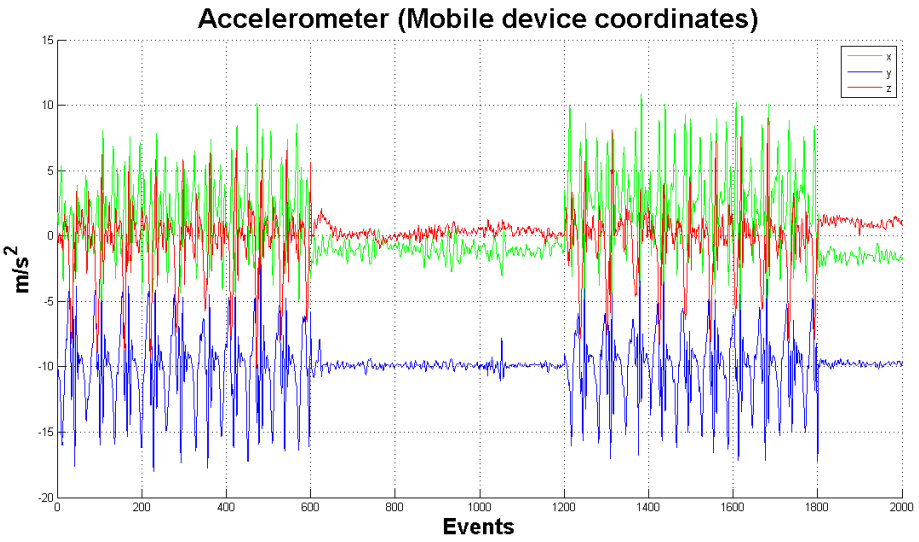

Figure 3.7: Sensing level: Device 3-axes accelerations.

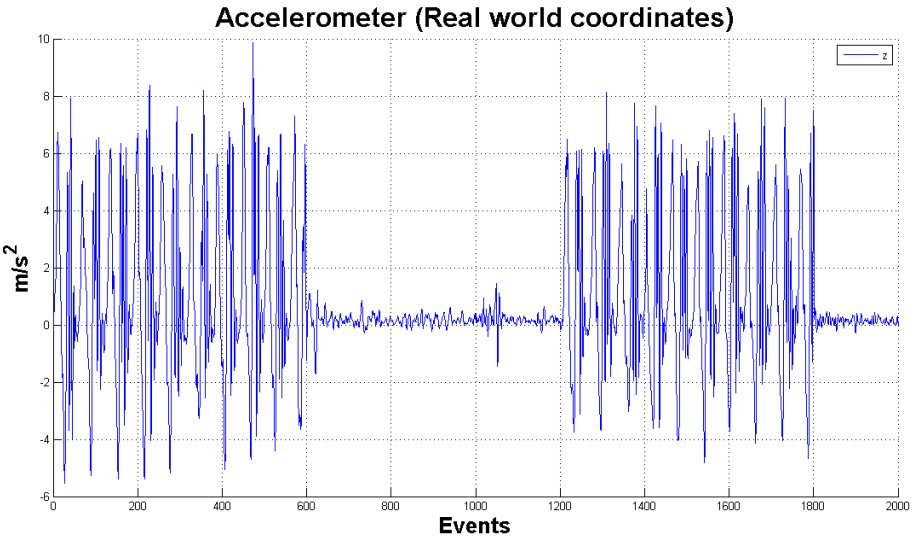

Figure 3.8: Real world vertical acceleration.

data is segmented using a sliding window with the defined 5 seconds of window size, shifted by 50% of overlapping.

3.1.5 Classification method

Once the features have been selected the next step is the classification. In the field of physical activity monitoring research, especially activity recognition, different classification approaches exist and yielded exceptional results. The benefit of using the data processing chain of Figure 3.1 is amongst others its modularity. This allows easily removing any module and replacing it with a different approach, thus different classifiers can easily be tested and compared to each other. In our case, we try to determine which features

Among the entire machine learning techniques we select a C4.5 decision tree. Particularly because the C4.5 presents several advantages over traditional supervised classification methods (Shoaib et al., 2015). In general, decision trees are fast in reasoning, which it is a crucial feature in real-time system. In addition, decision trees are tolerant to missing values, since a decision tree is defined as a classification procedure that recursively parts a dataset into smaller subdivisions and finally they are easily interpretable from developers because of the structure. For the training and evaluation of the three selected dataset within the preliminary studies of this subsection the Weka toolkit is used [1]. Weka (Waikato Environment for Knowledge Analysis) is a free machine learning software written in Java. It provides tools for analysing and understanding data, including the implementation of a large amount of data mining algorithms, and a graphical user interface for easy data manipulation and visualization.

Due to C4.5 is not implemented in Weka, the J48 decision tree is used which is the Weka version from the C4.5 decision tree algorithm. A detailed introduction into decision tree classification can be found in (Russell et al., 1995). C4.5 is a widely used algorithm to generate decision tree classifiers and is implemented in the Weka toolkit. Finally, the selected parameters to test each dataset are:

- Confidence Factor = 0.25
- Minimum number of object = 2
- unpruned = false
- Test-options = 10 folds Cross-validation

A commonly used evaluation technique to validate machine learning methods is k-fold cross-validation (CV). This technique randomly partitions a dataset into k equal size subsets. From these k subsets, k-1 are used as training data and the remaining subset as test data. This procedure is repeated k times, so that each of the k subsets are used exactly once as test data.

[1] Weka web page http://www.cs.waikato.ac.nz/ml/weka/

3.1.6 Performance Evaluation

This section presents and discusses the results of the 3 classification dataset, performed with the J48 classifier. Table 3.3 and graph 3.9 present the results in form of the 4 defined performance measures and also Tables 3.4, 3.5 and 3.6 present the confusion matrix of each dataset. In the rest of this section, some conclusions are drawn and discussed.

First of all, we will describe the commonly used performance measures, which are applied for creating the benchmark: precision, recall, F-measure and accuracy. The performance measures are defined generally and will be used for different classification problems. Besides, the tree size is a very important measure of the dataset to take into account, since the decision tree should be implemented in a real application, a bigger tree causes more energy consumption according to the increase of CPU cycles.

Assume that a confusion matrix is given by its entries $P_{i,j}$, where i refers to the rows (annotated classes), and j to the columns (recognized classes) of the matrix (See Tables 3.4, 3.5 and 3.6). Let S_i be the sum of all entries in the row i of the matrix (referring to the number of samples annotated as class i), and R_j the sum of all entries in the column j of the matrix (referring to the number of samples recognized as class j). Let N be the total number of samples in the confusion matrix. Let the classification problem represented in the confusion matrix have C classes: 1, 2,..., C. Using this notation, the performance measures precision and recall are defined as follows:

$$precision = \frac{1}{C} \sum_{i=1}^{C} \frac{p_{i,i}}{R_i} \tag{3.3}$$

$$recall = \frac{1}{C} \sum_{i=1}^{C} \frac{p_{i,i}}{S_i} \tag{3.4}$$

Therefore, precision can be interpreted as a measure of exactness (how reliable the results are in a class), while recall can be interpreted as a measure of completeness (how complete the results are of a class). Considering both the precision and the recall, F-measure is traditionally defined as the harmonic mean of them:

$$F - measure = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{3.5}$$

Finally, the measure accuracy is defined as the percentage of correctly classified samples out of all samples:

$$accuracy = \frac{1}{N}\sum_{i=1}^{C} p_{i,i} \tag{3.6}$$

Our main goal, however, was less to attain high recognition rates than to investigate to what extent the results of the dataset analysis generalize to the recognition results. Overall, the best performance was achieved with the statistical and the Spectrogram datasets (See Table 3.3 and Chart 3.9), however results are poorly accurate with the CWT one.

	Tree size	Accuracy	Precision	Recall	F-measure	MAE
CWT	17481	62.85%	43.86%	46.60%	45.19%	0.163
Spectrogram	2013	95.63%	95.88%	96.05%	95.97%	0.019
Statistical	1295	97.20%	95.91%	95.43%	95.67%	0.013

Table 3.3: Results obtained from every dataset using J48 decision tree.

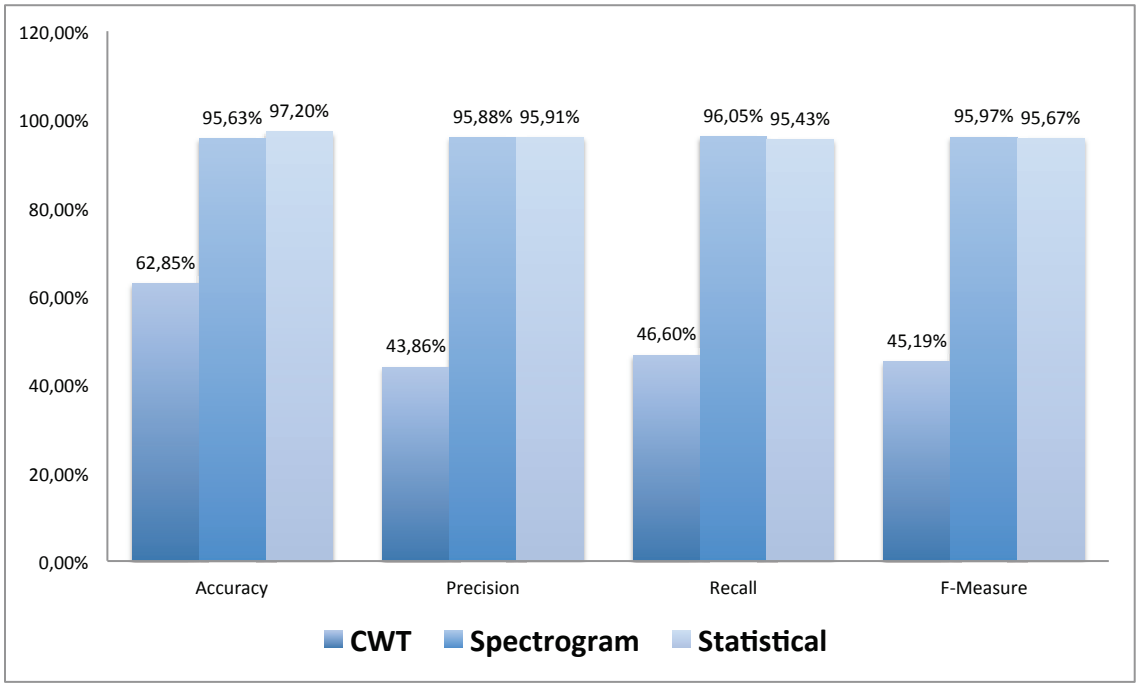

Figure 3.9: Accuracy, Precision, Recall and F-measure result comparison of every dataset (CWT, Spectrogram and Statistical).

Another general observation can be made when comparing the results is that the generated tree is smaller in statistical and Spectrogram however the statistical dataset present the smallest

one. How we previously said, a small decision tree is a good evaluation technique due to this significant performance difference. A smaller decision tree leads to an efficient algorithm in term on energy consumption and CPU cycles. Statistical vector is not only the most accurate dataset; otherwise it provides the smallest generated tree. Although, Spectrogram dataset present similar results to Statistical one, this dataset rely on the values of one signal (Vertical movement in the real world) and a cost pre-processing technique, which also in terms of energy consumption generates several drawbacks. Finally, CWT technique depicts the worst results of all studied dataset, besides it does not present any advantage over the other ones.

By studying confusion matrix (Tables 3.4, 3.5 and 3.6), results on the different dataset are generally in accordance with previous observations. For instance, the best dataset not only achieve approximately 97%, but misclassifications only appear into "neighbour" intensity classes. Concerning Spectrogram dataset, all performance measures significantly decreased compared to the Statistical one from 97.20%to 95.63%.

a	b	c	d	e	<–classified as
9462	3357	269	67	132	**a = Stopping**
3115	16957	2804	997	1852	**b = Walking**
302	3601	2025	658	1652	**c = Walking Fast**
114	1277	775	1100	2856	**d = Running**
169	2144	1439	1860	20266	**e = Running Fast**

Table 3.4: Confusion matrix of CWT dataset.

a	b	c	d	e	<–classified as
12815	472	0	0	0	**a = Stopping**
473	24592	640	20	0	**b = Walking**
0	667	72717	325	2	**c = Walking Fast**
0	23	395	5501	203	**d = Running**
0	0	4	212	25662	**e = Running Fast**

Table 3.5: Confusion matrix of Spectrogram dataset.

a	b	c	d	e	<–classified as
12979	308	0	0	0	**a = Stopping**
286	25047	392	0	0	**b = Walking**
0	360	7639	239	0	**c = Walking Fast**
0	1	247	5676	198	**d = Running**
0	0	0	187	25691	**e = Running Fast**

Table 3.6: Confusion matrix of Statistical dataset

Research	Classes	Sensors	Time(h)	Accuracy
(Miluzzo et al., 2008)	Still Walk Run	Accelerometer	4	78%
(Chon, 2011)	Still Walk Motor	Accelerometer Magnetometer Wifi GPS	28	91%
(Lester et al., 2006)	Still Walk Stairs up and down Riding elevator Brushing	GSM Wifi	7	84%
inContexto	Still Walk Run	Accelerometer	2	97%

Table 3.7: Comparative between most relevant works in activity recognition using inertial.

According to the presented results and that normally statistical features should be preferred for physical activity monitoring, we truly believe that statistical dataset is the best choice to implement an activity recognition module in inContexto. To conclude, the most relevant works in activity recognition with our technique are compare in the table 3.7.

3.1.7 Summary and Conclusion

This chapter presented a first approach to a data processing methods and classification algorithms for physical activity monitoring. A data processing chain is defined including pre-processing, segmentation, feature extraction and classification steps. For the classification step, different dataset are introduced and compared. First preliminary studies are carried out with a wide range of dataset (CWT, Spectrogram and Statistical Vector) using a J48 decision tree. Besides, the presented work further demonstrates that using a mobile phone providing with accelerometers is enough to infer physical actions that a person is taking place and also it is focus on the selected features which is an important field inside the Activity recognition systems.

We have seen that by defining different features and by comparing them to each other in terms of classification precision, one can obtain detailed information about how well a particular feature is suited for activity recognition. Our proposed measure of classification precision turned out to be a good indicator for the recognition performance of a feature. Overall, our results indicate that in contrast to an assumption that is sometimes implicitly made, there is neither a single feature nor a single window length that will perform best across all activities. However,

two of them present good result in terms of precision, recall, and accuracy.

We consider the best obtained solution the statistical vector which presents an overall accuracy of 97.20% well classify instances of 79250 different actions. This solution is a vector composed by: Energy, mean, standard deviation and correlation of each axes. However, next section will present a new method to infer people activity in order to reduce energy consumption in GPS available places.

3.2 Indoor and Outdoor classifier proposal

Once first experiment was over and clear which feature (Statistical vector) is the most suitable to infer people context a new proposal is presented in order to improve the overall performance of the system and also to include transportation modes to the activity recognition module. Moreover, in that case, the proposal aims to reduce the energy consumption and the number of uploads to the server in order to improve battery life and reduce broadband consumption and of course to enhance the overall performance of the activity recognition module. Image 3.10 shows a brief diagram of our new proposal implementing two classifiers depending on the GPS availability. Statistical dataset has been used to execute all the experiments.

Normally, users could be involved in two diverse situations, the first one is when the user is placed outdoors and the GPS is available and the second one, when the user is placed in a GPS unavailability places. These two situations are easily distinguishable using the mobile phone GPS value. On the one hand, when the user is outdoors, the GPS_{Speed} raw data contains a real value and on the other hand GPS_{Speed} value is set to null. For that reason, it is possible to split our activity recognition module in two, however, to implement another classifier which takes into account GPS_{Speed} is necessary. Although GPS consumes a lot of energy, the activity recognition precision improves significantly as we can see below.

Therefore, when the user is indoors in nearly impossible to perform motorized actions like driving a car, which our activities set, is reduced to Running, Walking, or Standing. On the contrary, when the user is outdoors motorized action is added to the activity set. Thus, taking into account context restrictions, Vehicles normally are used outside; so, we can improve in number of activities and accuracy our activity recognition module. Hence, now our activity recognition system should be composed by two different classifiers, one is executed just when the GPS signal is available (the user is outdoors, $action_{outdoors}$ classifier) and the second one is triggered just when the user is indoors ($action_{indoors}$ classifier) which allow us to turn off the GPS locks and preserve sensor battery life.

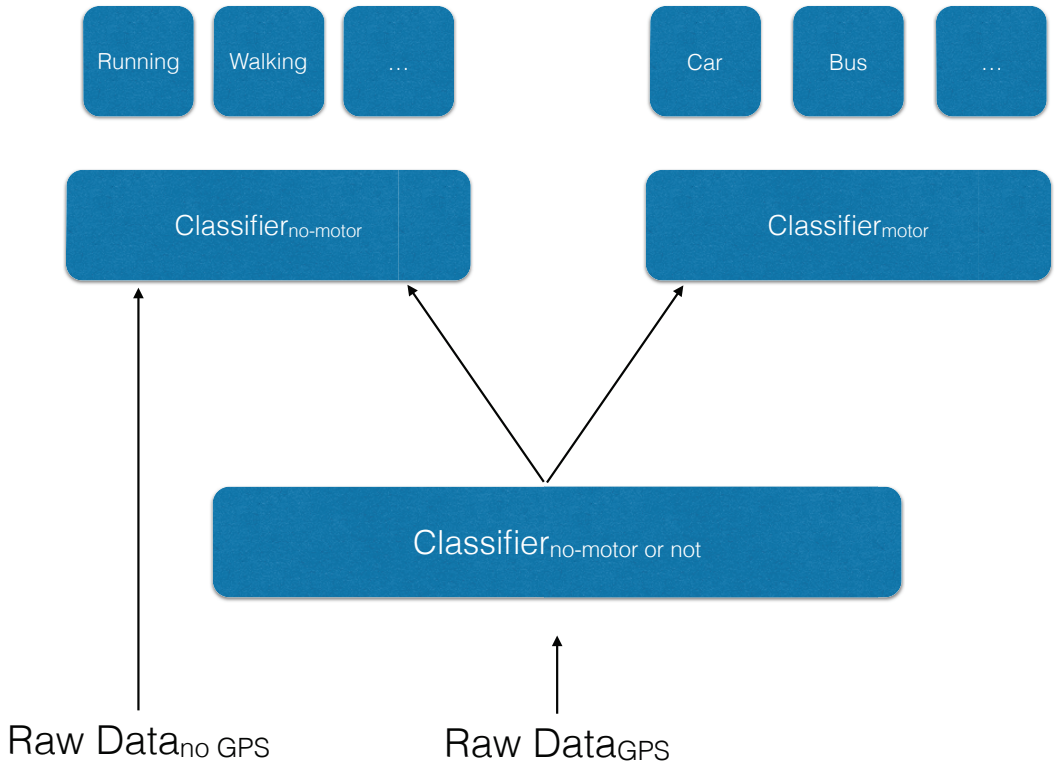

Figure 3.10: General scheme of Activity Recognition Module: Motorized and non-motorized classifiers.

3.2.1 MobilizeLabs Dataset

Since our dataset does not contain motorized actions, it was necessary to find an activity recognition dataset using smartphone accelerometer and GPS. Among all the existent dataset we believe that MobilizeLabs (MobilizeLabs, 2008) from the University of California (UCLA) is the most complete. The selected. The dataset is stored in a JSON format, which contains a list of data points for one user. Each point has the following fields:

- id: An identifier to distinguish users from each other.
- m: Activity mode, as classified from the data on the phone (still, walk, run, drive).
- l: A hashed location identifier. If this value remains constant, the user is in the same rough location (lat/lon with 2 decimal places).
- ts: Human-friendly time stamp.
- t: Timestamp in milliseconds since the epoch.

- data: Sensor data composed by:
  - sp: speed as provided by GPS, NaN if unknown.
  - wd: Wifi data.
  - ad: List of triaxial accelerometer samples within one second (up to 35 points on typical phones).

The dataset contains 235.768 labelled instances, however they are unbalanced since still actions have 10 times more than the other classes (walk, run and drive), for that reason we randomly selected 10% of the still instances. Finally our selected dataset is composed by:

	Still	**Walk**	**Run**	**Drive**	**Drive**
instances	19.775	10.457	5.447	21.667	57.348

Table 3.8: MobilizeLabs Dataset number of instances per class.

3.2.2 Performance Evaluation

This section presents and discusses the results of our new proposal of two classifiers depending on the user situation. We start by discussing the performance of the new proposal classifiers and then present the results from a detailed user study. We examine the classifiers performance based on a small-scale supervised experiment. As well as the previous study, we discuss classifiers accuracy, and the impact of our new proposal in the activity recognition module. As well as in the datasets study, the selected classifier is the J48 decision tree.

In order to evaluate the accuracy of our new proposal, first of all we have to start the training phase with our new dataset with the Drive class included. This new dataset contains four different classes: Still, Walk, Run and Drive and 57.384 instances distributed as Table 3.8 shows. In order to analyse the performance of the transportation mode classifier, three distinct metrics are employed: accuracy, precision, and recall. To test the new dataset, 10-fold cross validation is employed where the folds contain equal amounts of each activity and are made up off random continuous segments from the experiment data set. In general the achieved result are satisfactory compared to the previous dataset. Figure 3.11 shows the precision and recall values for the classifier, with an overall precision and recall levels both equal to 81.3%.

By developing our new proposal with two classifiers, our main goal is to overcome the results achieved with only one classifier 81.3%. The proposal consists on a previous classifier in charge of discriminate between motorized ($action_{motor}$) or not motorized action ($action_{no-motor}$).

```
Size of the tree :      2091

Time taken to build model: 7.4 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       46632               81.3141 %
Incorrectly Classified Instances     10716               18.6859 %
Kappa statistic                          0.7315
Mean absolute error                      0.1202
Root mean squared error                  0.2677
Relative absolute error                 34.536  %
Root relative squared error             64.1687 %
Coverage of cases (0.95 level)          95.8848 %
Mean rel. region size (0.95 level)      40.6178 %
Total Number of Instances            57348

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall  F-Measure  MCC    ROC Area  PRC Area  Class
                 0.797    0.096    0.814      0.797   0.805      0.705  0.921     0.821     still
                 0.864    0.039    0.83       0.864   0.847      0.812  0.952     0.817     walk
                 0.763    0.02     0.801      0.763   0.782      0.76   0.961     0.772     run
                 0.816    0.119    0.807      0.816   0.811      0.696  0.911     0.833     drive
Weighted Avg.    0.813    0.087    0.813      0.813   0.813      0.726  0.926     0.82

=== Confusion Matrix ===

     a     b     c     d   <-- classified as
 15762   112    12  3889 |     a = still
    74  9035  1013   335 |     b = walk
     4  1275  4158    10 |     c = run
  3527   459     6 17677 |     d = drive
```

Figure 3.11: MobilizeLabs dataset result using a J48 decision tree inferring Still, walk, run and drive classes.

Thus, now we have to create this new dataset changing still, walk and run class into no-motor class and the drive instances would be considered motor class. Nevertheless, a J48 decision tree with standard 10-fold CV is also applied as an evaluation technique in the experiments of this section for comparison reason. achieving the results that Figure 3.12 shows. In this case, the precision and recall achieve by the J48 decision tree are over 85%.

The last classifier that we have to train is the one whose activity classes are not motorized (Still, Walk and Run). As always, a J48 decision tree with standard 10-fold CV is also applied achieving over 92% in recall and precision benchmarks (See Figure 3.13).

To conclude, as we said before, this new proposal take into account that people normally is not doing motorized activities and non-motorized activities at the same time. Thus, we propose a previous classifier to determine if the user is doing a motorized ($action_{motor}$) or not motorized action ($action_{no-motor}$). Subsequently and depending on the previous result, two specific classifier are created, the first one is in charge of not motorized actions. This classifier is the same as we trained before in the section 3.1 and the second one is the one with Drive class included and trained in this section. The results of these seconds classifiers are summarized in the Table 3.9. Three motion states are detected by a J48 decision tree classification algorithm in the $action_{indoors}$ case. The results indicate that the simple actions

```
Size of the tree :      1293

Time taken to build model: 5.47 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       49157               85.717  %
Incorrectly Classified Instances      8191               14.283  %
Kappa statistic                          0.6969
Mean absolute error                      0.1894
Root mean squared error                  0.3292
Relative absolute error                 40.2811 %
Root relative squared error             67.9037 %
Coverage of cases (0.95 level)          97.6372 %
Mean rel. region size (0.95 level)      75.7481 %
Total Number of Instances            57348

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall  F-Measure  MCC    ROC Area  PRC Area  Class
                 0.882    0.183    0.888      0.882   0.885      0.697  0.909     0.917     no_motorized
                 0.817    0.118    0.808      0.817   0.812      0.697  0.909     0.834     motorized
Weighted Avg.    0.857    0.159    0.858      0.857   0.857      0.697  0.909     0.886

=== Confusion Matrix ===

     a     b   <-- classified as
 31462  4217 |     a = no_motorized
  3974 17695 |     b = motorized
```

Figure 3.12: MobilizeLabs dataset result using a J48 decision tree inferring Motor and no-motor classes.

```
Size of the tree :      721

Time taken to build model: 1.84 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       33147               92.9034 %
Incorrectly Classified Instances      2532                7.0966 %
Kappa statistic                          0.8783
Mean absolute error                      0.044
Root mean squared error                  0.1664
Relative absolute error                 15.0854 %
Root relative squared error             43.554  %
Coverage of cases (0.95 level)          97.7606 %
Mean rel. region size (0.95 level)      29.6316 %
Total Number of Instances            35679

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall  F-Measure  MCC    ROC Area  PRC Area  Class
                 0.99     0.007    0.995      0.99    0.992      0.982  0.995     0.995     still
                 0.897    0.057    0.868      0.897   0.882      0.832  0.964     0.883     walk
                 0.771    0.033    0.808      0.771   0.789      0.752  0.955     0.803     run
                 0        0        0          0       0          ?      ?         ?         drive
Weighted Avg.    0.929    0.025    0.929      0.929   0.929      0.903  0.98      0.933

=== Confusion Matrix ===

     a     b     c     d   <-- classified as
 19568   188    19     0 |     a = still
    98  9382   977     0 |     b = walk
     8  1242  4197     0 |     c = run
     0     0     0     0 |     d = drive
```

Figure 3.13: MobilizeLabs dataset final results using a J48 decision tree combined previous classifiers.

(Running, Walking, Standing) are recognized with an accuracy rate of up to 92.90% for the test cases employed in this study (Table 3.9). However, when the Motorized actions are included

the overall accuracy slightly decreases to 81.31%.

Classifier	Classes	Tree size	Accuracy	MAE
$action_{indoors}$	3	1295	92.90%	0.04
$action_{outdoors}$	4	2091	81.31%	0.12

Table 3.9: J48 tree accuracy in activity recognition.

Classes	Number of instances	Accuracy
$action_{motor}$	21,669	88.8%
$action_{no-motor}$	35,679	80.8%

Table 3.10: J48 tree accuracy discriminating motorized and no motorized actions.

This new classifier achieves a 85.71% of accuracy determining if the user is in a motorizes vehicle or is performing a physical activity. However, Table 3.10 shows that determine motorized activities is more accurate than no motorized ones. Then, we collect those instances that are classified as $action_{no-motor}$ and it makes a new classification using $action_{indoors}$ classifier in order to infer the concrete action the user is performing (Still, walk and run). By combining these two decision trees the overall performance of our system achieve 90.71%, which is much better than the accuracy obtained by $action_{outdoors}$ classifier at the first time.

3.3 Inferring user emotional context using Social Network Sites

This work is centred on determine emotional user information (happy, sad, fear, etc.). In that case, the main sensor is information shared by users on Social Networks Sites. The rest of the chapter details the design, implementation, and evaluation of the activity and emotional recognition system. The classification system is convenient for an individual yet reliable in accurately distinguishing between emotions and activities.

From a sociological point of view, narrative is inevitably emotionally structured and its analysis can be used as a method to study and determine emotions (Kleres, 2011). The emotion experience of human has a crucial narrative dimension. The key idea behind this is that emotion can be inferred and analysed if "who acts how to whom and what happens". In addition to this, external events are often regarded as the triggers of certain emotions. So in our work we attempt detect emotions by finding out why emotions are generated and how human readers feel them.

3.3.1 Building a data set for emotion analysis in Twitter

Due to Twitter restrictions is not possible to use a previous Twitter emotion dataset (Petrovic et al., 2010) to compare machine learning techniques. We had to create our own dataset to test NLP techniques in Spanish Tweets. In this section, it is described how we automatically created a labelled emotion dataset from Twitter SNS. Selected emotions were 6 (fear, anger, sadness, happiness, surprise, and disgust.) according to the Ekman research (Ekman and Friesen, 1978), also called six universal emotions.

Table 3.11: Matching between emotion hashtags with six universal emotions.

Emotion	Hashtag	Instances
Fear	#Miedo, #terror and #aprension	19.39%
Disgust	#Indignado, #asco and #repulsivo	23.74%
Sadness	#Triste, #sad and #infeliz	18.80%
Happiness	#Feliz, #happy and #contento/a	36.28%
Surprise	#Sorprendido, #sorprendida and #sorpersa	0.90%
Anger	#Furioso/a, #cabreado/a, #mosqueado/a and #enfadado/a	0.85%

We firstly collected at least 3 sets of emotion words for 6 different emotions (e.g., word "feliz" for emotion happiness) from existing psychology literature (Shaver et al., 1987). Subsequently, we retrieved tweets that have one of these emotion words as a hashtag (e.g, #feliz) using Twitter streaming API. Each collected tweet was automatically labeled with one emotion according to its hashtag (See Table 3.11).

In Computational linguistics and probability research fields, one of the best features to infer emotions using text is the ngram (Wilson et al., 2009). A n-gram is a continuous sequence of n items from a given sequence of text or speech. An n-gram could be any combination of letters. However, the items in question can be phonemes, syllables and letters, although using words give more information since they are more descriptive than the other ones. Nevertheless, to implement some text refinement techniques in order to reach a better emotion recognition system is extremely necessary. To create full sentiment analysis for a given question or topic requires many stages, including but not limited to:

1. Extraction of tweets using Twitter4J which is an unofficial Java library for the Twitter API.

2. Filtering out spam and irrelevant items from those tweets. The main filtering steps the we follow are:

- Anonymized username: We anonymize the user names since they do not provide relevant emotional information and also in the way to avoid malicious use of the data.
- Manual retweets (also known as "RT") are deleted because they do not give us relevant information.
- Tokenization is difficult in the social media domain, and good tokenization is absolutely crucial for overall system performance. Standard tokenizers, usually designed for newspapers or scientific publications, perform poorly because of the Twitter slang. However, we create a tokenizer which treats hashtags, @-replies, abbreviations, strings of punctuation and emoticons as tokens.
- Removing stopwords: We remove prepositions and conjunctions from the set of words since they do not provide enough meaning to each Tweet.
- Delete repeated characters: All repeated characters like spaces or repeated vowel are deleted in order to reduce the number of ngram in our dataset. Hence, words with the same meaning and slang differences (e.g. holaaaaaaa -> hola).
- Negation form: "no" word is attached to a word which follows it. For example, a sentence "No quiero ir" will form two words: "no+quiero", "ir". Such procedure allows to us improve the accuracy of the classification since the negation change completely the meaning of the sentence since it plays a special role in an opinion and sentiment expression (Pak and Paroubek, 2010).

3. Identifying subjective tweets. A set of filtering heuristics was developed to select the most valuable tweets.

   - We kept only the tweets with the emotion hashtags at the end. In previous works was proved that the most relevant words are at the end of a Tweet (De Choudhury et al., 2012).
   - We discarded tweets which have less than five tokens, since they may not provide sufficient context to infer emotions.
   - URL del Tweets which contains URL links since the relevant information is stored in the link (e.g. http://example.com).

After the filtering process concluded, we totally collected 21,991 relevant tweets from a period spanning December 28th 2012 until January 8th 2012.

3.3.2 Emotion Classification Results

In order to test the results, we selected Weka's implementation from Multinomial Naive Bayes (MNB) since it provides good performance with a large-scale dataset. Taking into account microblogging text characteristics which maximum text length is 140 characters, we chose small n values. Hence, we decided to compare results between different values of n: unigrams (n=1), Bigrams (n=2), Trigrams(n=3) and the Unigrams and Bigrams (n=1, 2) combination, see Table 3.12. The experimental results show that unigrams yields better performance than using unigrams alone (Figure 3.14). While the number of ngrams are increasing the accuracy decreases (from 65.12% to 36.40%). The number of ngrams and the accuracy show that unigrams provides the best performance to infer twitter user emotion.

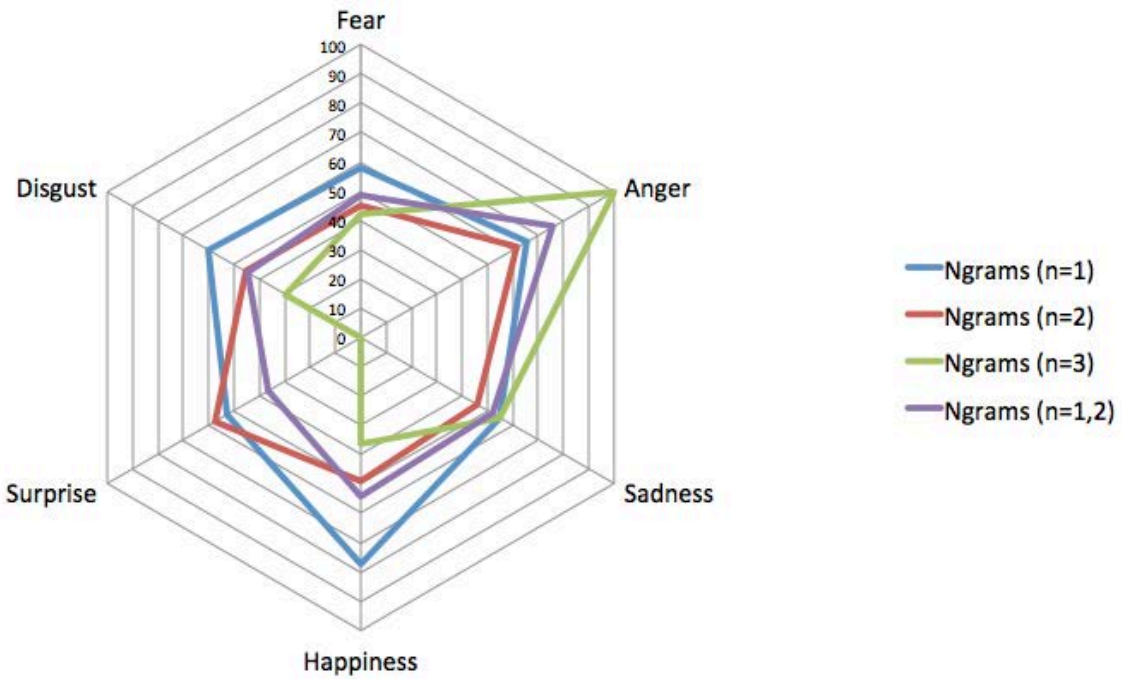

Figure 3.14: Representation of the accuracy obtained for each technique (ngram, bigram, trigram and the combination) according the six basis emotions.

We train classifiers with unigram features for each emotion class using Multinomial Naive Bayes (MNB) for predicting the emotion category of the sentences in our corpus. MNB provides good performance with a large-scale dataset and has previously given good performance in sentiment classification experiments.

According to the described features above, one of the best method to analyze emotions in microblogging context is using N-grams. The most common sizes for n are 2 (bigrams), 3 (trigrams) and 4 (four-grams) because unigrams are too narrow a unit of analysis. In each

Table 3.12: Machine learning accuracy (ngrams).

Features	Number of ngrams	Accuracy
ngram(n=1)	2264	65.12%
ngram(n=2)	1381	47.64%
ngram(n=3)	164	36.40%
ngram(n=1,2)	3645	49.72%

experiment, we represent every sentence by a features vector indicating if a ngrams appears in the sentence or not. It is made a Boolean feature for each n-gram, which is set to true if and only if the n-gram is present in the tweet.

Our main goal for these experiments is to compare different features in NLP using Spanish Twitter Corpus. Taking into account microblogging text characteristics which maximum text length is 140 characters, we chose small n values. Hence, we decided to compare results between different values of n: Unigrams (n=1), Bigrams (n=2), Trigrams(n=3) and the Unigrams and Bigrams (n=1, 2) combination.

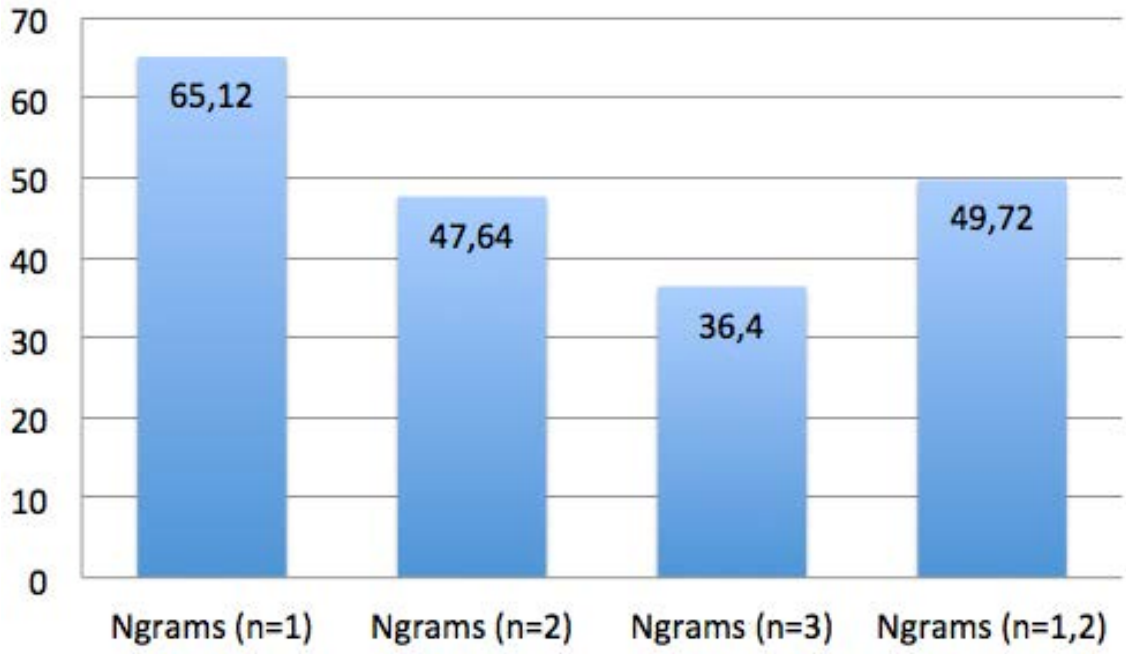

Figure 3.15: Ngram technique total accuracy.

In the first experiment, we use only corpus based unigram features. We obtain high precision values for all emotion classes (as shown in Table 3.12). Besides, Table it shows the overall performance of MNB classifier (trained with all tweets) on each emotion category. Our experimental results show that unigrams yields better performance than using unigrams alone.

While the number of ngrams is increasing the accuracy decreases (from 65.12% to 36.40%). The number of ngrams and the accuracy show that unigrams provides the best performance to infer twitter user emotion. This validates our previous premise since we consider unigrams can help learn lexical distributions well using short sentences (AS Tweets) in order to accurately predict human emotion categories.

It is important to highlight that the three most popular emotions, which account for 36.28% of all tweets, the classifier achieves precisions of over 75% (Unigrams). On the contrary, performance declines can be seen on less popular emotions (i.e., Surprise and Anger), which consist of 1.75% of all the tweets in our dataset. The precisions of these two emotion categories are relatively high (with lowest precision of 58.1%). Interestingly, how is decreasing continuously the performance on the evaluation data comes from using bigger n-grams together with the lexicon features and the microblogging features.

Specifically, combining unigrams and bigrams decrease the accuracy to 49.72%. Hence, further incorporation of trigrams was not implemented due to bad result for using one of them alone. As well as existing works on NLP emotion recognition (Pang et al., 2002) using unigrams alone is better than applying either bigrams, trigrams or a combination of unigrams and bigrams.

3.4 Summary and Conclusion

In this chapter, we investigate two different kind of people context, the first one is related to physical context and the other one is about emotions. We afford both from different points of view, in the emotional case; we use SNS as sensors in order to collect people information. On the other hand, activity context was inferred from embedded smartphone sensors.

First of all, this chapter presents a study comparing three different techniques to infer activity context using a J48 decision tree. The study is focus on giving to inContexto architecture the possibility of inferring activity context (Running, walking, driving, etc.) from accelerometer data. Overall, the presented work further demonstrates that using a mobile phone providing with accelerometers is enough to infer actions that user is doing. The best given solution obtained an overall accuracy of 97.20% classifying instances of 79250 different actions. This solution is a vector composed by energy, mean, standard deviation,

The second study trays to find a way of detecting the sentiment of Twitter messages in a Spanish corpus. We evaluate the training data with labels derived from hashtags. Moreover, we culled Spanish emotion tweets covering 6 emotions (fear, anger, sadness, happiness, surprise,

and disgust) categories for automatic emotion identification. The experimental results show that the feature of unigrams presents better performance than, bigrams, trigrams and the combination of both of them. We achieved the highest accuracy of 65.12% with is more or less the same accuracy that other researchers have obtained in previous works using English Twitter datasets. Considering future works are to increase the accuracy of the classification, we should discard common n-grams, measured using Chi-squared. For example taking the top 1,000 n-grams. Besides, it is possible to reduce misspellings and grammatical error in order to unify ngrams.

4

inContexto: a Framework to Collect, Infer and Share People Context

THIS chapter describes inContexto in an architectural way. InContexto is a new architecture for building mobile context-aware system in ubiquitous scenarios with rich social data. It allows a combination of different social sources and types of mobile clients in order to obtain contextual information to enhance people's life. Thus, users can easily access the system employing their smartphone or tablet; while at the same time, user social information can be analysed coming from social networks or third parties applications that also store significant social or behavioural data.

inContexto exploits off-the-shelf sensor-enabled mobile phones to automatically infer people's activities (e.g., dancing at a party with friends) and also collecting information through social network portals such as Facebook or Twitter. The main goals of our architecture are: (i) Collection, storage, analysis, and share of the user context information, (ii) Plug-and-play support for a wide variety of sensory devices, (iii) Privacy preservation of individuals sharing their data, and (v) Easy application development.

The theoretical approach leads to a client-server architecture service through Rest technology to determine the user's and their relatives activities. On the one hand, the client is composed by the individuals and their surroundings, collecting and sharing context information. On the other hand, the server gathers all this information from multiple clients and improves it applying machine learning techniques.

This chapter is organized in the following way: Section 4.1 presents some technical considerations to take into account in order to implement a framework to infer, create and store people context information. Then, section 4.2 defines the implemented architecture in the portable devices and also the backend which is in charge of storing people context.

4.1 Design Considerations

Before describing the implementation of the inContexto application on the smartphone and backend servers, we first discuss the system development challenges encountered when implementing an application such as inContexto on the phone. Mainly, there are two important characteristics to take into account before to develop a sensing application using smartphones.

The main factor in terms of system effectiveness is energy consumption on client side where mobile device users are sensitive about the battery usage for installed third-party applications. Another important factor to take into account in energy consumption problem is sending data through connectivity interfaces: The application should be improved by data traffic efficiency and also should balance extraction ratio between sensor data and full comprehensive information. Extraction ratio is dependable on granularity of provided information and sampling frequency of sensed captured data. All these factors clearly impact in several aspects of the architectural design; hence, they must be taken into account. The last factor to consider is user acceptance. All these applications collect personal information (position, movements, emotions, etc.) and store them. Normally, people are very reluctant to yield their actions to third party apps.

4.1.1 Battery management

In this subsection we briefly discuss barriers to adoption of context smartphone apps in terms of battery and power as well as some workarounds to mitigate those barriers. Certain hardware limitations present in mobile sensing computing are likely to be overcome in near future, for example computing capabilities nowadays is less important since smartphones already host multi-core processors and gigabytes of memory. However, battery related issues, on the other hand, still represent a significant burden to continuous sensing. Distributed mobile sensing brings new, socially aware, applications, but with additional challenges of data management. Nevertheless social-aware is not exempted of battery consumption drawbacks. While moving the computation tasks over to the shared pervasive computing devices helps conserve the in-network energy, repeated communications to convey the samples to the pervasive computing devices depletes the sensors battery quickly. Hence, smartphone context applications should balance these operations in order to preserve the energy consumption.

In general, most of sensing applications do not require a continuous sampling to achieve their purposes. In sensing tasks aiming to observe the motion of humans, data is useful if it allows analysing people motion over time in a certain physical space. Therefore, reducing the interfaces' use, through periodic sampling, it is possible to reduce power consumption.

This approach involves setting a sampling period for each interface, in order to maintain the efficiency of data collection. Defining the sampling periods requires the understanding of which factors may influence the data collection efficiency. The interfaces differ in characteristics, such as energy consumption, access method and operation range. In chapter 3 was discussed the differences between continuous collecting information or periodic sampling.

Considering the Internet interface connection characteristics, the operation range of GSM is much higher than WiFi's, therefore the sampling period for GSM should be longer, on the contrary the energy consumed by the Wi-Fi is much higher. Other aspect to take into account is the time that a scan takes to be completed. A Bluetooth scan, for example, takes about 13 seconds to be completed; therefore using a sampling period shorter than 13 seconds is inefficient.

The battery's lifetime depends on its capacity, the smartphone usage level and the hardware consumption. In (Pendão et al., 2014) a good study is presented in terms of energy consumption in Smartphone motion recognition. For example, the work shows energy consumption comparative with the main smartphone interfaces. The energy consumption values for Wi-Fi and GPS presented in table 4.1 were collected from a presentation made by Jeff Sharkey, in 2009 at Google IO conference (Sharkey, 2009).

	Wi-Fi	**GPS**	**Bluetooth**
Idle	12 mA	0 mA	6 mA
Active	275 mA	85 mA	50 mA

Table 4.1: Smartphone consumptions per interface.

Taking into account the energy consumption values shown in table 4.1, the impact of continuous and periodic sampling on the smartphone's battery life may be been calculated. The objective is to estimate only the connection interfaces consumption, therefore the CPU's consumption discarded, and also the LCD consumption has been considered zero since it should be turned off during the process. Selecting a battery with 1200 mAh of capacity, the estimation suggests that continuous sampling completely drains the battery in nearly 3 hours, confirming that a continuous sampling is not a good solution in terms of usability. However, comparing with periodic sampling, we observe a significant energy consumption reduction, which is reflected in the battery lifetime is extended at least 3 times more than continuous sampling.

According to the tested results from discovery (Mazumder, 2011), the energy consumption of localization sensors is the most effective in Global System for Mobile communication (GSM) cell bases evaluation where energy costs lowest than $<$20 mJ followed by WLAN around 545 mJ and finally Global Positioning System (GPS) sensor is consuming much more $>$1,424 mJ.

However, the precision of localization sensors is inversely proportional. Moreover, availability is another key factor in positioning system. Therefore, an efficient solution may order the systems according to energy efficient factor.

To sum up a solution the proposed solution for reducing energy consumption in mobile sensing applications using inertial sensors embedded in smartphones is to implement a synchronous service for collecting sensor information periodically (every five minutes) and use most efficient location system if it is possible.

4.1.2 Users

In general people have some reservations in sharing their personal information (Location, activity, etc) to aps. Smartphones are directly linked to the personal lives of its users and are constantly pointing their location, compromising their privacy. In some cases, users of a collaborative sensing system allow their personal device to serve the purposes of the system without receiving any type of reward and abdicating some of their privacy. However, to maximize and maintain the number of participants over time, it becomes necessary to define a motivating reward model that will achieve and retain an adequate number of participants. The type of reward can vary from financial rewards to the access to a service. If the reward is attractive, the user may be willing to waive their privacy. Social networking and other services use this type of strategy.

Kapadia et al. m(Kapadia et al., 2009) addresses these problems by describing challenges and discussing possible solutions. Shin et al. presents the AnonySense (Shin et al., 2011) system, that authors describe as a privacy-aware system for creating applications based on opportunistic collaborative sensing in personal mobile devices. However, the truth is that the privacy of users is practically impossible to guarantee, as shown by recent results from some researches. Movement patterns of an individual can be identified directly through GPS samples, or indirectly, through the Wi-Fi access points and GSM base stations (Gonzalez et al., 2008).

Moreover, there is also a technical challenge that compromises the adherence of people to mobile sensing applications: the use of some interfaces of mobile devices in sensing tasks, such as the GPS, the Wi-Fi, and Bluetooth, involves high energy consumption, with a strong impact on the autonomy of the device. Although users can charge devices frequently, the high-energy consumption of sensing applications for smartphones immediately condemns them to failure. Users can abdicate their privacy without major reservations as long as the reward is attractive, but not the autonomy of their devices. Nowadays, users use continuously smartphones, so if an application dramatically reduces the autonomy of the device, making it unavailable or forcing the user to charge it constantly, it will be uninstalled.

4.2 inContexto: Distributed Architecture

In the previous chapter, we presented various frameworks, specifications, techniques, etc. for building inContexto architecture systems. These technologies were classified in terms of four abstraction levels, namely, Raw Data and Features, Simple Actions, Social Actions and Action Context, according to the degree of refinement of the managed values. In the picture 4.1 is show the architectural design of inContexto as we previously describe in the second chapter.

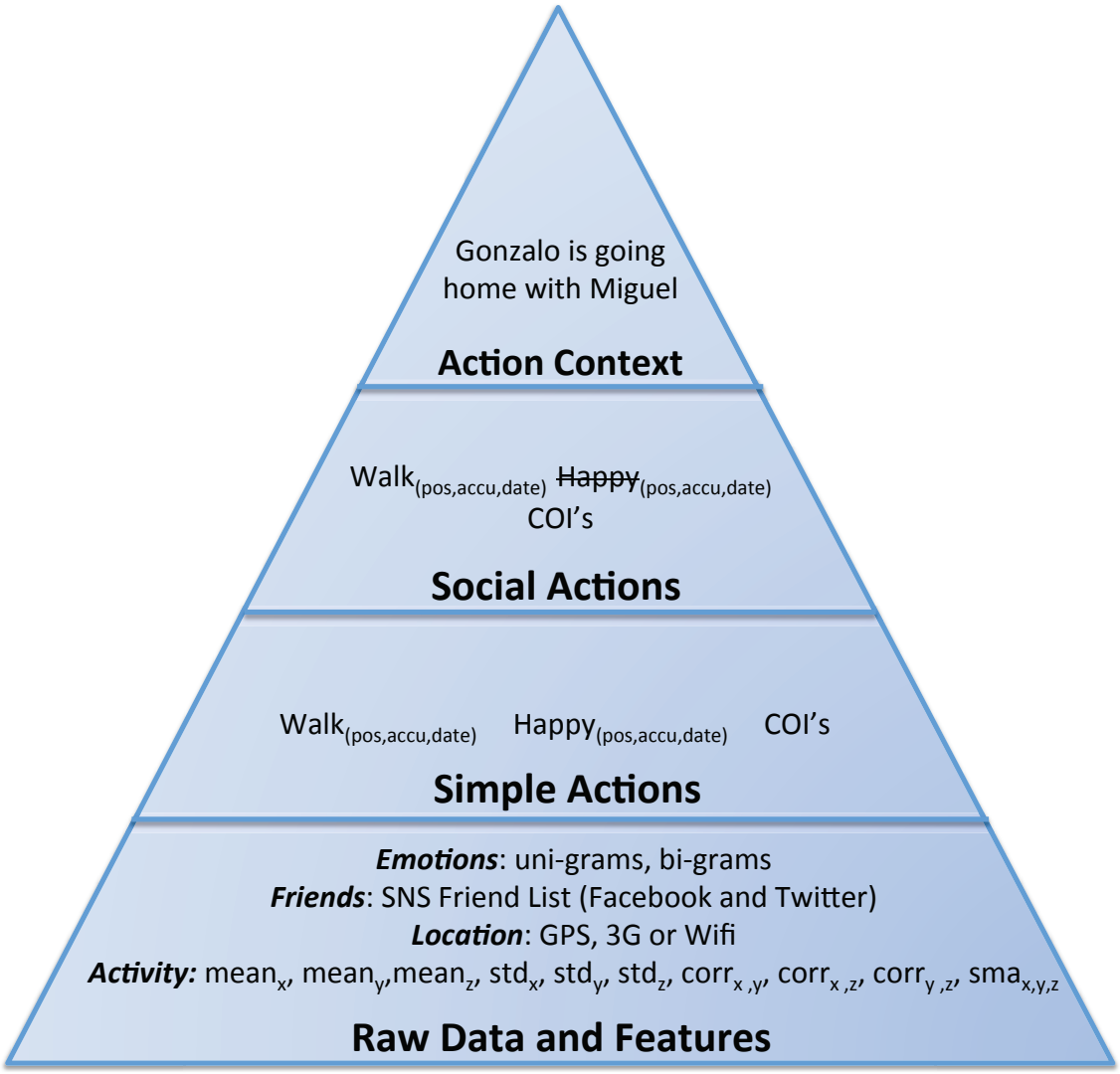

Figure 4.1: Layered and pyramid based design of inContexto architecture.

The implementation of inContexto might be started from scratch, and rely on these techniques. Although such a straightforward approach to the problem could be timesaving in the short term, it would probably turn out to be expensive in the long run. The reason for

this is that software maintenance, support, and extension costs often soar if the system is very complex, and if incompatible, immature, or undocumented technologies are used.

The following objectives must be achieved by the architecture. Most of them correspond to non-functional requirements of context-aware systems, which are requirements that are not related to what the system has to, but rather to how it is done. Unfortunately, these features are not orthogonal and are often in conflict with each other. First, the architecture must be adaptable to a variety of situations, namely, different domains, interaction patterns between elements, and technologies. Two other features that a context aware architecture must provide are extensibility and robustness. Extensibility may be required in diverse dimensions, for instance, in the number and type of system users or in the amount of resources available. A robust system guarantees that it will be operative most of the time. The more critical the task to be resolved, the greater the effort we must made to build a reliable system.

Given these requirements, we propose an architecture based on the client-server paradigm. The reason is that the client-server paradigm has proved to be capable of supporting the development of distributed systems in several domains with different communication schemes. Moreover, client-server systems are scalable, adaptable to different requirements, and robust. The client-server technique is one the most frequently used architectures in distributed computer. Basically, it suggests a simple intercommunication schema between two clearly distinguished entities: clients and servers. Clients are programs that perform minimal processing, in our case retrieving user context information, and require few computational resources. However, a server performs all the actions computationally cost. Hence, clients delegate most of the tasks to servers, which run on more powerful platforms, and are responsible for satisfying client requests.

Our contribution provides an abstract architecture for intelligent mobile systems that overcomes some of the issues previously described. This architecture relies on the client-server and service-oriented architectures. Services are modelled as resources that every single client is able to consume. inContexto application and system support consists of a software suite running on Samsung Galaxy with Android OS installed and a backend infrastructure hosted on server machines. To sum up, the software installed on the phones performs the following operations: sensing, feature extraction of the raw sensed data, showing user information, and the upload of the collected features to the backend servers. First of all, it has to be emphasized that is not clear what architectural components should run on the smartphone and which ones should run on the backend server. Workload can be balanced between clients and servers in such a way that clients may need servers only to carry out certain complex tasks such as database management.

In Section 4.1 some interest situations to take into consideration before to decide are

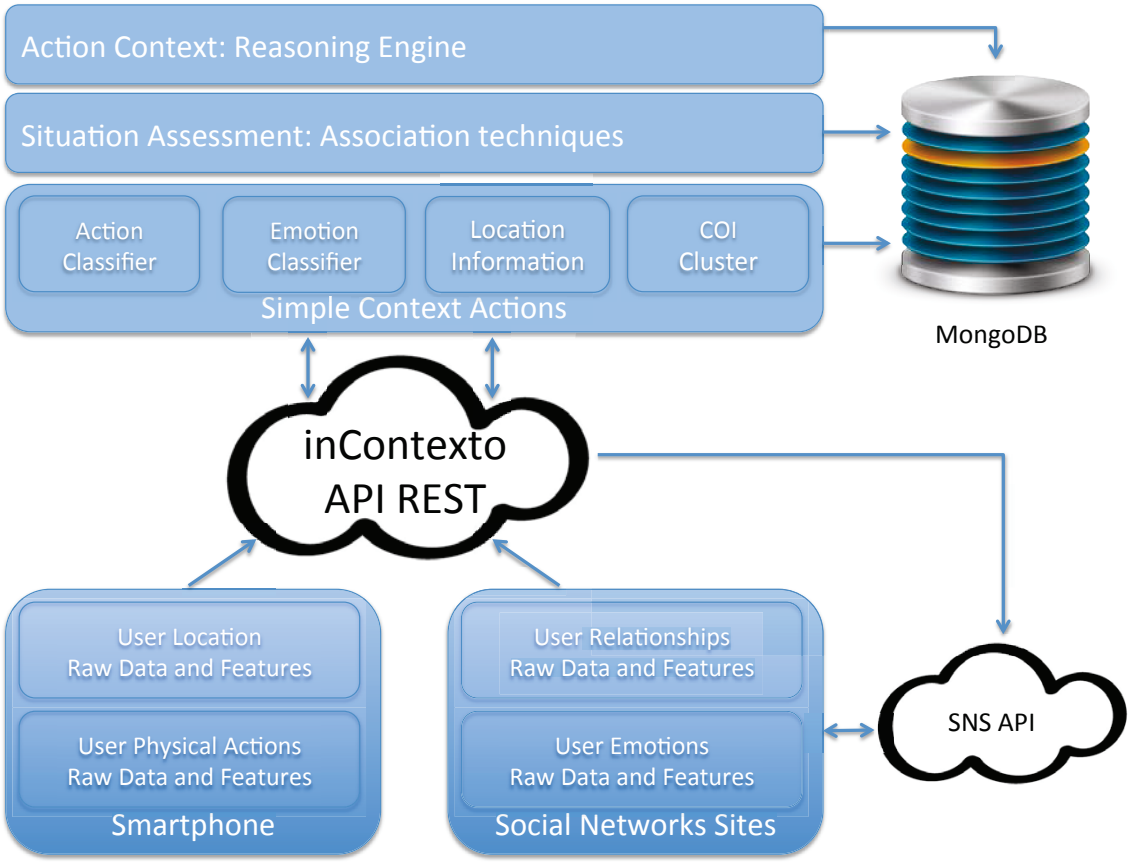

Figure 4.2: inContexto architecture overview splitted on each application (Backend, Front-end and Smartphone application).

described. Energy consumption, network availability and user privacy force us to develop just the Level 0 (Raw Data and Features) in the smartphone and of course the rest of the levels are implemented on the cloud. In level 1, 2 and 3 are implemented computationally cost techniques (Machine learning algorithms, fusion techniques, etc.), which makes difficult to implement on the mobile phone. Therefore, this architecture is split in two modules: Smartphone module and backend module. Conceptually, there are a sensing client, which may be installed on the user smartphone (Physical activity) or on the cloud (Emotional activity) and periodically polls on-board sensors (physical or emotional) via the available network connection (Wi-Fi or cellular) to the backend server, which receive all the user context information for an analysis and storage. A general overview of inContexto architecture is depicted in Figure 4.2.

This illustration 4.2 shows four different components. In the bottom part is placed the front-end modules, in this case physical activity recognition implemented on the smartphone and the emotion activity recognition implemented on the cloud. The front-end module collects user context information related to user activity, moods, relationships and Location. On the top part

of the figure is depicted the backend infrastructure which receives user context information from API REST and joint them generating a more accurate user information description. Besides, this module it is placed the storage module, which is mainly a database storage.

4.2.1 inContexto Client Devices: Collecting context

At the bottom of the context pyramid is placed client context devices which aim to collect information surround them. Although we always describe smartphones and background daemon to collect user information, there should be thousand of devices that may collect relevant information. For example, meteorological stations, heart rate monitors, air quality station, etc. As it is described at chapter 2, all the clients devices must provide context information with a specific geometry or situation and of course if this observed property is related to a person, the person inContexto identification.

The client devices are responsible for: i) operating the surroundings presence inference over the sensor data by locally running the collection services ii) communicating the raw information to the backend service; iii) in smartphone cases displaying the user's and their buddies sensing presence (activity, social context, location), the privacy configurations, and various other interface views that allow, for example, a user to post short messages on their preferred social network account. At this point, the develop client services are three: user devices (mainly smartphones), background daemons and surroundings software.

According to the presented architecture in 2, we develop three different inferring services, the first one a personal monitoring service using the smartphone, a background service in order to infer emotional state, and finally a custom devices to infer surrounding information. In particular, we will extend our previous approach for activity identification in two ways. One is to use multimodal sensing and the other is to expand the set of activities identified from a basic set (e.g. walking, typing) to more complex activities (e.g. cooking, brushing teeth).

4.2.1.1 Smartphone Software

The smartphone client aims to collect all the user context information related to physical activities. Although this client may be improved attaching new sensors like heart rate monitors or blood pressure one in our case we just use the embedded sensors. The smartphone collect periodically sensor information and bulk to the server side. Our approach is based on a duty-cycle design point that minimizes sampling while maintaining the application's responsiveness. This design strategy allows inContexto to operate as near to real-time as possible; that is, some

system delay is introduced before a person's sensing presence is updated on the backend servers. In the case of the current implementation the introduced delay varies according to the type of presence being inferred. A delay between two consecutive data improves the overall energy efficiency.

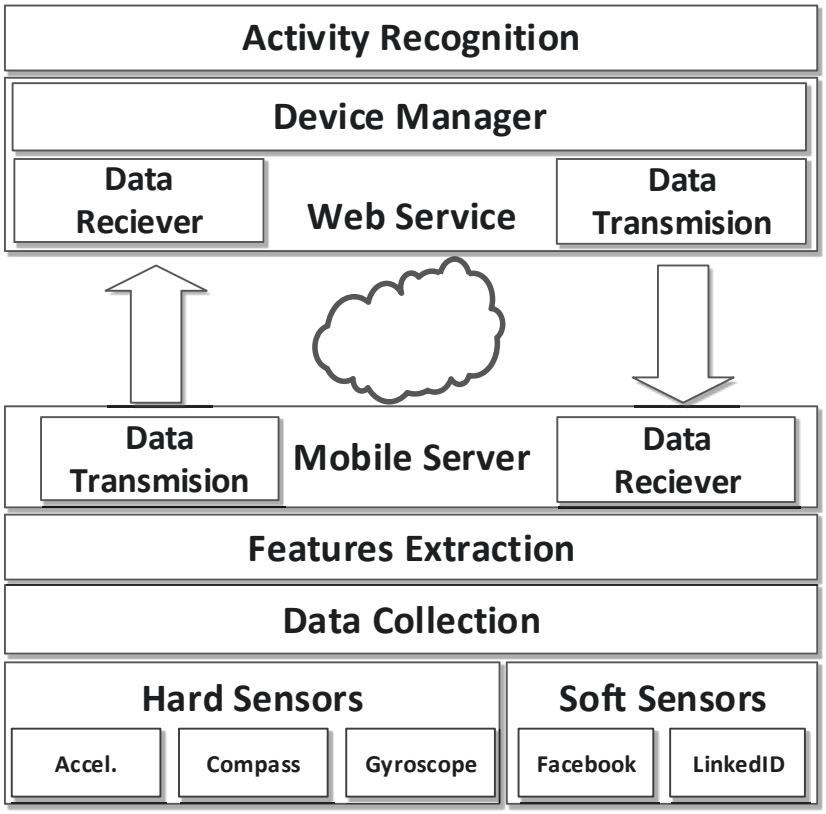

Figure 4.3: inContexto level 0 and level 1 architectural representation.

The proposed middleware has been implemented in Android to enable the dynamic management of location resources and access control to assist continuous monitoring and long running proactive location-based services. Android is a Linux-based, open-source mobile-phone platform. Most core phone functionality is implemented as applications running on top of a customized middleware (See 4.3). Applications are written in Java and compiled to a custom bytecode format known as Dalvik EXecutable (DEX). Each application executes within its own Dalvik VM interpreter instance. Each instance executes as a unique UNIX user identity to isolate applications within the Linux platform.

Recall that the Android SDK provides a dedicated API for accessing embedded smartphone sensors as location information, accelerometer raw data, video cameras and so on. All this

API are designed in a generic way, such that the same API can be used to retrieve location information from different localization techniques by using the location provider that uses the GPS receiver embedded in the device or the network connectivity facilities to get a location estimate.

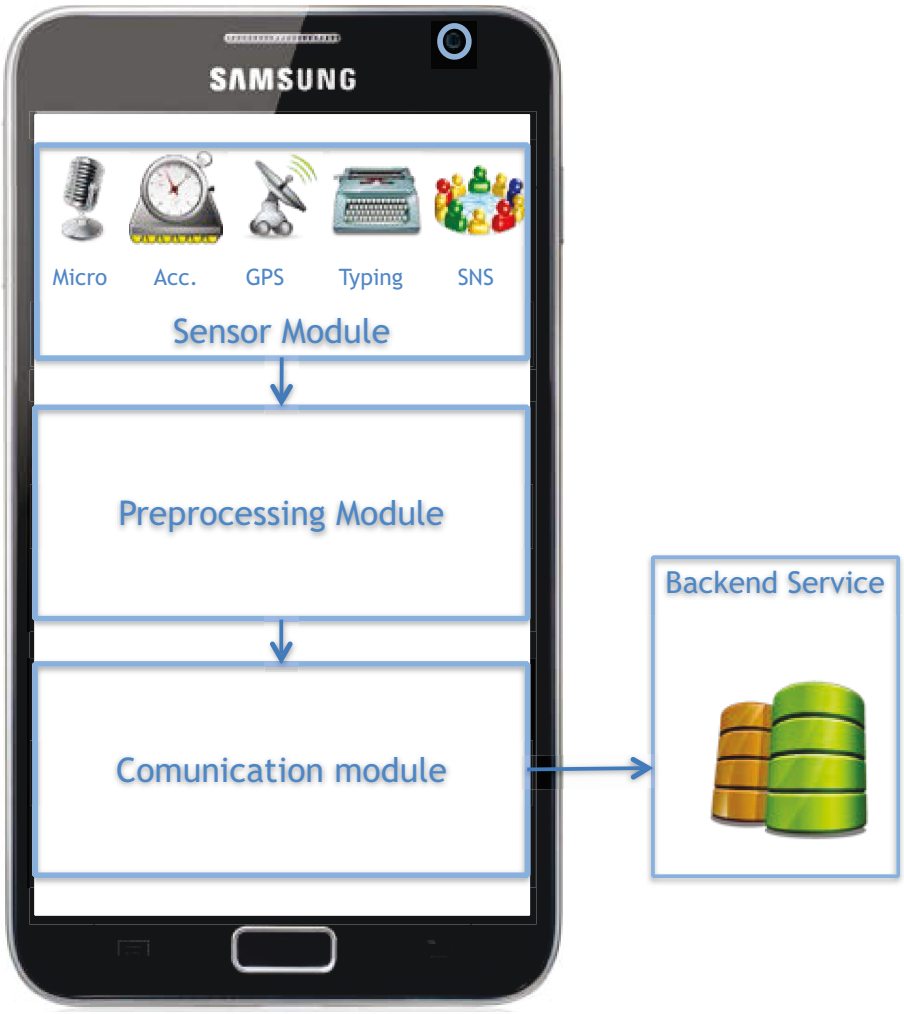

Figure 4.4: inContexto smartphone architecture.

Figure 4.4 shows the inContexto software architecture for the Android OS. The phone architecture comprises the following software components.

- *Sensor Module*: This component connects through the sensory API to retrieve the sensor information data byte stream. The byte stream is sent to the preprocessing module in order to reduce the amount of sent data, as discussed previously. These events are received through a socket from the event detector daemon and it is stored locally until

the next upload session. On the other hand, capturing GPS data takes a different algorithm. The implementation supplies a callback method that is periodically called by the Location API to provide the geographical location of the phone. The GPS coordinates are uploaded to the backend servers at the same time with the other information following a GeoJSON format, but before they are pre-processed as the rest of data. This component is responsible for orchestrating the underlying sensing components. The pre-processing module starts, stops, and monitors the sensor clients and GPS daemon to guarantee the proper operation of the system.

- *Preprocessing Module*: This module aims to pre-process raw data from the sensor module in order to reduce the amount of information sent. Depending on the sampling frequency, this module receive from the sensor module user raw data information After that, this module stores the raw processed sensed data records to be sent to the cloud framework. When the pre-processing of raw data records is performed, the data records are discarded; hence none of the sampled data persists on the phone. This is particularly important to preserve the integrity of the data and the privacy of the person since none of the raw sensed data is ever transferred to the backend.

- *Communication Module*: This component is responsible for establishing connections to the backend servers in an opportunistic way, depending on radio link availability, which can be either Wi-Fi or mobile Internet connection (Cellular). If this action is not possible to accomplish, the communication module will store user information until the connectivity come back. It also uploads the primitives from local storage and tears down the connection after the data is transferred.

4.2.1.2 Background services

This services aims to collect information from an external source that provides an API. This information is not under the individual supervision since once they grant inContexto to access to certain service, this will executed periodically until they revoke the permission. For example, inferring user emotional state as we saw in Chapter 3 needs a background process to access periodically to Social Networks Sites in order to collect user information. Moreover, there is so many interesting public API to feed our framework.

4.2.2 inContexto Backend Server

As we explained later inContexto framework is able to infer and share context information from various consumers to use. However what happens to data that is not being consumed by a consumer or a third party application? What if a consumer wants to perform queries based on historical data? These questions can be answered by the introduction of a data logging service. In other words, a component that can collect sensing data and then provide services based on this data. The server side system is responsible for:

i Storing user context information and allowing other inContexto users (customers) restricted access to this data;

ii publishing this sensor presence information to third party social network such as Twitter and Facebook;

iii performing last levels of inContexto architecture which aims to classify information in order to get more specific one;

iv performing routine user registration and select which classifiers wants; and

v the collection of statistics about user behaviour both on the client and backend side of the system.

Server logic can be implemented with multiple developer languages (J2EE, .NET, nodeJS and so on). Web applications are usually structured in two or three levels, decoupling the backend storage (e.g. databases) and the presentation layer. The interface of a web application can be oriented to human users or automatic procedures. Web applications are systems accessible throughout the Internet using the HTTP protocol. For human users, HTML pages with a suitable format are created in the server and sent to the client; for automatic procedures, there is an entry point managed by a callback procedure, i.e. a Web Service, enabled to accept JSON requests. Web Services are used in machine-to-machine communications, i.e. communication acts where human participation is reduced. Web Services have succeeded as means to implement cooperative distributed systems.

An alternative approach using Web Services is the REST model (REpresentational State Transfer), which proposes a simple architecture based on ad-hoc XML or JSON request-response messages and APIs. This model is preferred when the power and the formality of the Web Services protocol stack are not required. inContexto offers RESTful APIs to register user information and listeners. User information is a continuous query registered with relevant

information from permitted users to upload data that feeds inContexto architecture and it allows to listeners to generate new information. The collection back-end has been designed exploiting MongoDB with GeoJSON extension (for geospatial processing) as database. This layer of the system architecture is in charge of the spatial and temporal data aggregation.

A cloud-based storage is considered due to its potential to store a large-scale number of sensing applications concurrently to support ubiquitous individual and community context analysis. The cloud database has a table to store the location information (identified as Location). It is used to store and to identify the latitude and longitude coordinates from the mobile application. Moreover, the time at which the location points are acquired in the smartphone (in YYYY-MM-DD HH:MM:SS format) is stored. All this information is considered sensible, hence, it should be stored meticulously and deny access to unauthorized applications or people (Section 4.3 shows the implemented methods to preserve user privacy).

In each machine is running a Debian distribution with an Apache Web Server, Apache Tomcat and a mongodb service. The inContexto backend is a Web application which is implemented on the Digital Ocean cloud infrastructure [1], is comprised by a series of different virtual machine images.

4.2.2.1 API Rest

After some years, Internet architects have found an alternative method for building web services in the form of Representational State Transfer (REST). REST is a style of software architecture for distributed hypermedia systems such as the World Wide Web. The term Representational State Transfer was introduced and defined in 2000 by Roy Fielding in his doctoral dissertation(Fielding, 2000), (Fielding and Taylor, 2002).

REST-style architectures consist of clients and servers. Clients initiate requests to servers; servers process requests and return appropriate responses. Requests and responses are built around the transfer of representations of resources. A resource can be essentially any coherent and meaningful concept that may be addressed. Although REST was initially described in the context of HTTP, is not limited to that protocol. RESTful architectures can be based on other Application Layer protocols if they already provide a rich and uniform vocabulary for applications based on the transfer of meaningful representational state. RESTful applications maximize the use of the pre-existing, well-defined interface and other built-in capabilities provided by the chosen network protocol, and minimize the addition of new application-specific features on top of it. In a REST environment, clients are not concerned with data storage, which

[1] https://cloud.digitalocean.com

remains internal to each server, so that the portability of client code is improved. Servers are not concerned with the user interface or user state, so that servers can be simpler and more scalable. Servers and clients may also be replaced and developed independently, as long as the interface is not altered. Finally, servers are able to temporarily extend or customize the functionality of a client by transferring logic to it that it can execute.

Client-Server architecture has been evolved to Service Oriented architectures (SOAs), which, to some extent, may be considered an enhancement of the client-server paradigm. Functionalities in a SOA are provided by services, which are stateless facilities that can be accessed remotely. Services are usually implemented with Web Services, a standard from the World Wide Web Consortium. Web Services also separate clients and servers, which have very different roles in the architecture. Nevertheless, in some situations a client may become a server (and vice versa), in such a way that each component of the architecture can alternatively request and provide information. This pattern is known as Peer-to-peer (P2P), and depicts a scenario where groups of loosely coupled equally responsible entities communicate directly to accomplish an objective.

In order to provide an efficient and scalable method to access user context information, the backend infrastructure was made following REpresentational State Transfer (REST) an architectural model presented by Roy Fielding in his doctoral dissertation (Fielding, 2000). REST is stateless client-server architecture over HTTP protocol whose principal advantages are: Scalability, Generality of interfaces and Independent deployment. REST architectures rely on the powerful of HTTP protocol. Clients initiate requests to servers; servers process requests and return appropriate responses. Since every petition is requests to a HTTP server, is possible to response several connections at the same time, also the bottleneck is reduced thanks to the HTTP servers features. REST frameworks create a representation of URI resources. Every resource is expressed uniquely through URI, creating in that way, a uniform interface that provides a common access to the user context information. In other case, there are three different resources:

- User: User resources describe the user information that is stored $inContexto_{id}$, SNS_{id}, $user_{name}$, $user_{birth}$ and $user_{pic}$. Besides, there are implemented all the possible GET, DELETE and POST.

- Measurement: The measurement resource is a representation of the L0 features. Depending on the source sensor (Smartphone or SNS), the information is different but it follows the same structure: $inContexto_{id}$ and $user_{features}$.

- Action: This is the final result (Action Context), the given information is the user and the concrete actions that is performing.

The normal concept of a resource is a web page. Your web browser might want to retrieve it using the HTTP method GET, for example. However, some resources actually store information or create additional resources. For example, if you have a blog, you might use the HTTP method POST to create a new blog entry. The "thing" to which your browser issued the POST method is a resource (some type of blog "factory") and the new blog entry is another resource.

Operation	HTTP Method	Response Code
Insert or update	POST	200 CREATED: The data was successfully inserted or updated into the database. 400 BAD REQUEST: The data insert or update operation did not complete successfully.
Obtain	GET	200 OK: The response body and content-type are retrieved from a previous insert or update operation. 404 NOT FOUND: The specified key is not present in the database. 400 BAD REQUEST: The appliance was unable to process the request.
Delete	DELETE	200 NO CONTENT: The entry was deleted from the database. 400 BAD REQUEST: The appliance was unable to process the request.

Table 4.2: Operations with equivalent HTTP methods or verb and response code definitions.

The important idea to understand, though, is that REST actually encourages the use of more HTTP methods than what most web browser support (at least at the time of this writing in 2008). Most web browsers only support two HTTP methods, namely GET and POST. Yet, HTTP also defines the methods PUT, DELETE, OPTIONS, HEAD, TRACE, and CONNECT. The REST Binding Component supports the GET, PUT, POST, DELETE, and HEAD operations. Next table (Table 4.2) present which operation are implemented in inContexto.

Hence, according to the architecture depicted in Chapter 2, inContexto API Server is split in four layers, one per each context level. Moreover, the API provides a subscription interfaces for upper levels (From 1 to 3) in order to connect inContexto with third part applications. In the next tables are shown the created interfaces to develop inContexto backend server, it is mandatory to add the base url according to the site where is deployed.

This first level (Table 4.3 provides four different interfaces, the first one and more important is a POST method to store user context information into the application (inContexto). Besides, it provides two different ways to access this information, the first one is demanding all the user information stored on the cloud, and on the other hand the method whose response is the last value of a specific user property. Last interfaces are just a summary of the stored information about the user.

URL	Method	Description
/level0/userID	POST	This endpoint is the most important one in the architecture since it provides the interface to add information context from a user. The JSON object contains the information described in the Chapter 2
/level0/userID	GET	Interface to recover last vector information from the user corresponding to userID. It provides all the information stored to the user in level 0.
/level0/userID/property	GET	As well as the last method this interface provides the last value of a specific property from a user.
/level0/userID/offering	GET	This interface return all the user observed properties and its measurement unit.

Table 4.3: Level 0 inContexto interfaces.

In the second level presents more or less the same functionality than the previous one. However, in this case there are developed a couple more interfaces in order to register inference methods. This new method just provides information about the algorithms that the user is subscribed and another interface to subscribe a user to a specific one.

Level 2 presents five different interfaces in the inContexto architecture. First of all, two of them are developed for setting and getting relevant user context information at this level. On the other hand, three different methods are implemented in order to configure the filtering process for each user.

Last level is developed with the same interfaces that the level 1, however the provided information is more accurate and specific than the previous one. To sum up, there are seven interfaces in charge to accomplish the given functionality.

As it was described in previous sections, this architecture provides the a way to evolve user context information from raw data (signals, words, etc) to more specific information (moods,

URL	Method	Description
/level1/userID	POST	Every single inference method should insert new context information using this interface. As it was explained before the JSON object contains the position, time and new context information.
/level1/userID	GET	Interface to recover last vector information from the user corresponding to userID. It provides all the information stored to the user in level 1.
/level1/userID/property	GET	As well as the last method this interface provides the last value of a specific property from a user.
/level1/userID/offering	GET	As well as the last level, this interface provides a summary of the stored information (Observed property) about the user in the level 1.
/level1/info	GET	Method in charge to return information about all the implemented algorithms at this level.
/level1/userID/method	POST	Interface that add a new inference method to a user. If the user does not have the needed observed properties to infer this new information, the method will not be triggered.
/level1/userID/info	GET	Interface that return the inference methods that the user is subscribed.

Table 4.4: Level 1 inContexto interfaces.

actions, etc). Moreover, inContexto should allow to third part application take part in the whole process seeding or feeding the architecture.

4.3 inContexto: Security and Privacy

Privacy and security are very sensitive issues for sensing applications. Solutions to protect user's privacy have been proposed however they are focused only on the stored data. However, inContexto sends information through Internet to the backend infrastructure. Communication between frontend and backend is probably the most important point according to privacy and security problem. A secure and robust authentication system is mandatory in order to send sensory information to the cloud preserving user privacy. Security is a very important property

URL	Method	Description
/level2/userID	POST	Once the filtering process was finish, the information is inserted at this level following the JSON format exaplined before.
/level2/userID	GET	Interface to recover last vector information from the user corresponding to userID. It provides all the information stored to the user in level 2.
/level2/userID/property	GET	As well as the last method this interface provides the last value of a specific property from a user in the level 2.
/level2/userID/time/time	POST	Method to determine the new value for the time filtering process.
/level2/userID/location/radious	POST	Interface wich allow to change the radious to filter SA from the level 2.
/level2/userID/time/coi	POST	Interface which define which community of interest should be take into account in order to find new realtions.
/level2/userID/info	GET	Method that return the values of the three different filters.

Table 4.5: Level 2 inContexto interfaces.

that deserves a special comment. Security entails user authentication, integrity of the data, and encryption of the communications among other tasks. Sensing application and specifically, inContexto architecture, must be secure, but our architecture does not tackle this matter in depth because a complete study of it is outside the scope of this work.

4.3.1 AWS Authentication Scheme

Authentication is the process of proving your identity to the system. Identity is an important factor in inContexto access control decisions. Requests are allowed or denied in part based on the identity of the requester. As a user, you will perform requests that invoke these privileges so you will need to prove your identity to the system by authenticating your requests. In the literature are implemented different solutions, from using complex standards as OAuth to passing credentials over HTTP. In the next lines will study those techniques most used by the industry, in concrete, OAuth and AWS protocol, which are the most used authentication methods. The Amazon Web Service (AWS) uses a custom HTTP scheme based on a keyed-HMAC (Hash Message Authentication Code) for authentication.

URL	Method	Description
/level3/userID	POST	Every single inference method should insert new context information using this interface. As it was explained before the JSON object contains the position, time and new context information.
/level3/userID	GET	Interface to recover last vector information from the user corresponding to userID. It provides all the information stored to the user in level 3.
/level3/userID/property	GET	As well as the last method this interface provides the last value of a specific property from a user.
/level3/userID/offering	GET	As well as the last level, this interface provides a summary of the stored information (Observed property) about the user in the level 3.
/level3/info	GET	Method in charge to return information about all the implemented algorithms at this level.
/level3/userID/method	POST	Interface that add a new inference method to a user. If the user does not have the needed observed properties to infer this new information, the method will not be triggered.
/level3/userID/info	GET	Interface that return the inference methods that the user is subscribed.

Table 4.6: Level 3 inContexto interfaces.

To authenticate an HTTP request, you first concatenate selected elements of the request to form a string. You then use your AWS Secret Access Key to calculate the HMAC of that string. Informally, we call this process "signing the request," and we call the output of the HMAC algorithm the "signature" because it simulates the security properties of a real signature. Finally, you add this signature as a parameter of the request, using the syntax described in this section. The following task list describes the basic process for authentication.

1. To create a string based on specific information in the request. The selected string is the UTF-8 encoded value of the Date header in the request (e.g., Thu, 17 May 2012 19:37:58 GMT). It is mandatory to include either the Date header in order to check the integrity of the petition. The format you use for the header value must be one of the full date formats specified in RFC 2616, section 3.3.1 (Fielding et al., 2009).

2. To calculate a signature using your AWS Secret Access Key, the string from task 1, and using an RFC 2104-compliant HMAC-SHA1 hash algorithm and finally convert the resulting value to base64.

```
String: Thu, 17 May 2012 17:08:48 GMT
Secret Access Key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Base64-encoded signature: 4cP3hCesdCQJ1jP11111YSu0g=EXAMPLE
```

3. Include the signature in the request and send the request to the server using HTTP. To pass the signature to the server, it is mandatory to include it as part of the standard HTTP Authorization header. Since the server will check if the authorization is correct is included both the signature and the access token in the header using the following format:

   `Authorization: ICT <Access Token>:<Signature>`

   Finally, the following lines shows an example of REST authentication using AWS scheme:

```
GET /measure/1519820224 HTTP/1.1
Host: incontexto.eu01.aws.af.cm
Date: Mon, 03 Mar 2013 19:37:58 +0000
Authorization: INC AKIAIOSFODNN7EXAMPLE:frJIUN8DYpKDtOLCwo//yllqDzg=
```

4. When the system receives an authenticated request, it fetches the Access Token that you claim to have, and uses it in the same way to compute a "signature" for the message it received. If the two signatures match, it is accepted and processed the request. Otherwise, the request is rejected.

The main drawbacks presented by using credentials (Username and Password) is that they are usually reused across many sites, hence, they are much easier to intercept, if compromised on one site, it could be compromised for all sites, etc. Conversely, access tokens are securely randomly generated strings over 40 characters long and have significantly greater entropy, and are much harder for attackers to compromise, since normal passwords length is about 8-13 characters. This is virtually unnoticeable to the end-user logging in to an application, but it makes it almost impossible for an attacker using brute force to compromise a single password that is trying millions of potential passwords at once.

Finally, access Token can be revoked and generated a new one periodically or removing keys if you think one might have been compromised. Users, who are already signed on inContexto,

do not need to re-enter their credentials when granting authorization to any API request. Not having to enter credentials frequently is a real benefit for mobile users. Besides, Token passing authentication method is a stateless algorithm; hence, it is not necessary to keep the connection with the server.

4.3.2 inContexto authentication and identification

Regarding the state of the art in security in REST web services, it has been implemented a lo-gin and authentication scheme following the main advantages which present AWS scheme and third party OAuth systems. inContexto security scheme involves two different steps, user identification y user authentication.

Firstly, an inContexto user has to register in inContexto using Facebook Platform (FP) in order to identify each inContexto user with one Facebook user. Hence, Facebook is considered an identity provider. Facebook Platform leverages OAuth 2.0 for authentication and authorization process. It is a connect service which let third-party application to retrieve SNS features (Ko et al., 2010). Besides, FP is an open standard that describes how users can be authenticated in a decentralized manner, obviating the need for services to provide their own ad-hoc systems and allowing users to consolidate their digital identities (Recordon and Reed, 2006).

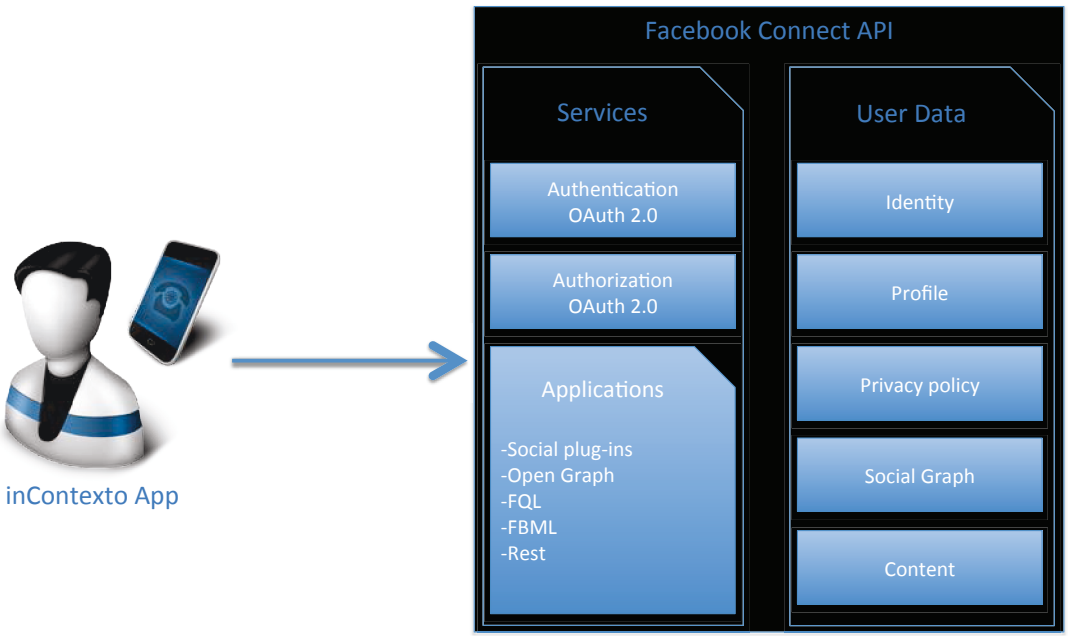

Figure 4.5: Facebook Platform connect architecture.

When a user clicks the lo-gin button, one dialog appears depending on the previous user

state. If the user is not already logged into Facebook, the login pop-up dialog appears. When the user authorizes inContexto in Facebook Connect, Facebook open a session for this user and generates a callback to the inContexto app. However, if the user is logged in and has already authorized Connect for inContexto, Facebook reopen the session for this user and provides a callback to the main inContexto window. Facebook callback response contains Facebook identification and the user access token. Once the Facebook session is open, Facebook allows inContexto to access to the API and collects user basic information profile: user name, date of birth and personal email and also Facebook relationships or friends.

Finally, user personal identification data from Facebook is collected and sent to the inContexto platform in order to create a new inContexto user account. Adding a new user in inContexto is very easy following the inContexto REST API specifications described before. In that case, it provides a POST HTTP operation to the user resource 4.3.2 (Note that the Facebook user token length is 40 characters, however, it is reduced in order to preserve the integrity).

```
POST /user/{facebookID} HTTP/1.1
Host: incontexto.eu01.aws.af.cm
Date: Mon, 03 Mar 2013 19:37:58 +0000
Body: {"name":"Gonzalo",
"firstName":"Gonzalo",
"lastName":"Blazquez Gil",
"birthday":"21/06/1983",
"userToken":"86bd12ce8fd6067e2f15d9eb8ffc88bb"}
```

Once inContexto server receive the new user request 4.3.2 checks if the user is previously registered on the system. Otherwise, inContexto security manager generates a opaque and globally unique userID (Instead of using "1234", it is used "f6cd3259f9a39c9732"). Using not-sequential numbers decreases fusking attack (Guess the next id number), and prevents contention issues by being able to disperse UUID generation to any server. Since, the userID is going to be sent through Internet, it should be generated by a byte array encoded using 62 'url-safe' characters. This allows IDs to be safely used in URLs without having to worry about encoding problems. Subsequently, also inContexto security module generates a new access token to allow users access to the REST API. This token is randomly generated using the java.security.SecureRandom java API class which is designed to be cryptographically secure. In practice, this means that the generator has the following properties:

- Given only a number produced by the number generator makes impossible to predict previous and future id numbers.
- The produced numbers contain no known biases.
- The number generator has a large period, based on the 160-bit SHA1 hash function, the period is 2160.
- the number generator can seed itself at any position within that period with equal probability.

Although SecureRandom produced numbers are impossible to predict, periodically, it is recommended that to eliminate the existing SecureRandom and create a new one with a new seed.

Now, when the user is properly registered every single request to inContexto architecture should be authenticate. in our case, the following method is Amazon Web Services guidelines, updating the authentication header in the HTTP request every.

```
POST /user/{facebookID} HTTP/1.1
Host: incontexto.eu01.aws.af.cm
Date: Mon, 03 Mar 2013 19:37:58 +0000
Authorization: INC " + accessTokenId + ":"
+ base64(hmac-sha1(VERB + "\n"
+ URL + "\n"
+ DATE + "\n",
token))
```

4.4 Context Care: An inContexto study case

According to the financial crisis situation around the world, reduced budgets is a real challenge for governments. In this case, to freeze or reduce the healthcare budget is a priority while at the same time the service improves its quality. Thereby, Ambient Intelligent (AmI) applications aim to contribute to reduce costs and offer better and more efficient services. Anytime and anywhere assistance requires several underlying mechanisms and tools (López et al., 2010): ranging from wireless-enabled monitoring, location systems that permit us to identify where they are located in case of needed.

There are many potential uses for AmI in Business management scenarios (Cook et al., 2009), however, ContextCare architecture is focused on human resources monitoring. Specially, hospitals and nursing homes scenarios where it possible to increase the efficiency of their services by monitoring patient's health, progress, and routines through the analysis of their activities, decreasing budgets.

Activity recognition is traditionally carried out through video systems like those described in (Cilla et al., 2011) and (Gómez-Romero et al., 2011). However, video activity recognition systems present few problems like: huge computational cost, speedy bandwidth to transmit information, complex techniques to interpret video data and also people to decide if a action is right or wrong is necessary. Recent researches in activity recognition shows that Micro-Electro-Mechanical Systems (MEMS) is becoming another way to face with activity recognition problem (Avci et al., 2010; Gil et al., 2012a).

Therefore, we present ContextCare as an event-driven Semi-supervised video surveillance system that involves the use of smartphones and visual sensors. ContextCare monitoring system rely on two different architectures: inContexto (Gil et al., 2011a) which monitors user activity and location and video surveillance system (Bustamante et al., 2011) which uses inContexto information to focus on an anomalous situation reducing the time spend by security personnel in front of a screen.

4.4.1 ContextCare Scenario Model

Our interaction scenario considers a space populated with video cameras responsible of monitoring people. The architecture is implemented following a event-driven model where patient context (event) triggers actions depending on a given condition (ECA rules). An ECA rules is divided in three different parts: the*event* is the signal that triggers a set of rules; the *condition* which if is satisfied makes the execution of the rule to continue and finally, the *action* defines the execution flow of a process. ECA Rules are created by experts (security personnel or emergency services) and represent patient emergency situations. ContextCare architecture contains three different applications to deliver the global functionality:

- *Video surveillance system:* has been developed in C++ modules (as described in the previous works section) and it was evaluated in previous works (Bustamante et al., 2011). Summarizing, the frame-rate obtained is the same as the video sensor provides (25 FPS).
- *Patient smartphone application:* The technological platform in the current prototype is an Android smartphone. Patient Smartphone app mainly consist in a monitoring service

implementing inContexto architecture which collect patient sentinel data and establish user activity and location. Besides, is also able to reason with the raw sensor data to identify higher-level information including patient activities or even though beat rate, temperature, etc.

- *Security mobile terminal (SMT):* This application provides the necessary tools to access patients information. When a event triggers a rule, SMT receive a message which contains: patient profile, video tracking about patient situation, patient location and the event that was happened. Moreover, it is also provided, which allows the emergency services to evaluate the most recent medical details obtained from sensors, performs new measurements, and communicates with the caretakers.

Consider a patient in a nursing home wearing a smartphone with the normal sensors and blood pressure sensor connected. inContexto architecture captures every sensor data and generates patient context information. This information or event may be dispatched a rule (ECA rules) depending on the ECA rules condition. For example, an unusual blood pressure level triggers one rule and provokes a set of actions to be followed, sending a message to his surveillance system or even his family members, physician, emergency services, friends or colleagues.

The final goals of ContextCare study case is making a real time video surveillance and monitoring system in a health care scenario where the main actions to track are: People activity (Run, Stand, Laying) a fall detection system and finally health care monitoring as heart beat rate.

4.4.2 Proposed Architecture: ContextCare

The main functionality of the ContextCare Surveillance Platform (See figure 4.6) is to connect seamlessly patients and the health professionals, reporting to the latter alerts and health measurements obtained from the patients. Besides, ContextCare is designed to manage video surveillance flow and determine which emergency member is the most suitable to look after the patient taking into account patient health context.

Below we will focus on describe every single inContexto level since as we previously define; inContexto architecture modules are different depending on the specific situation. ContextCare study case provides to patients a smartphone in order to track their movements and situation.

ContextCare level 0 (Raw data and features) is a sensing module which continuously collects relevant information about the patient. This level is implemented on an Android OS device

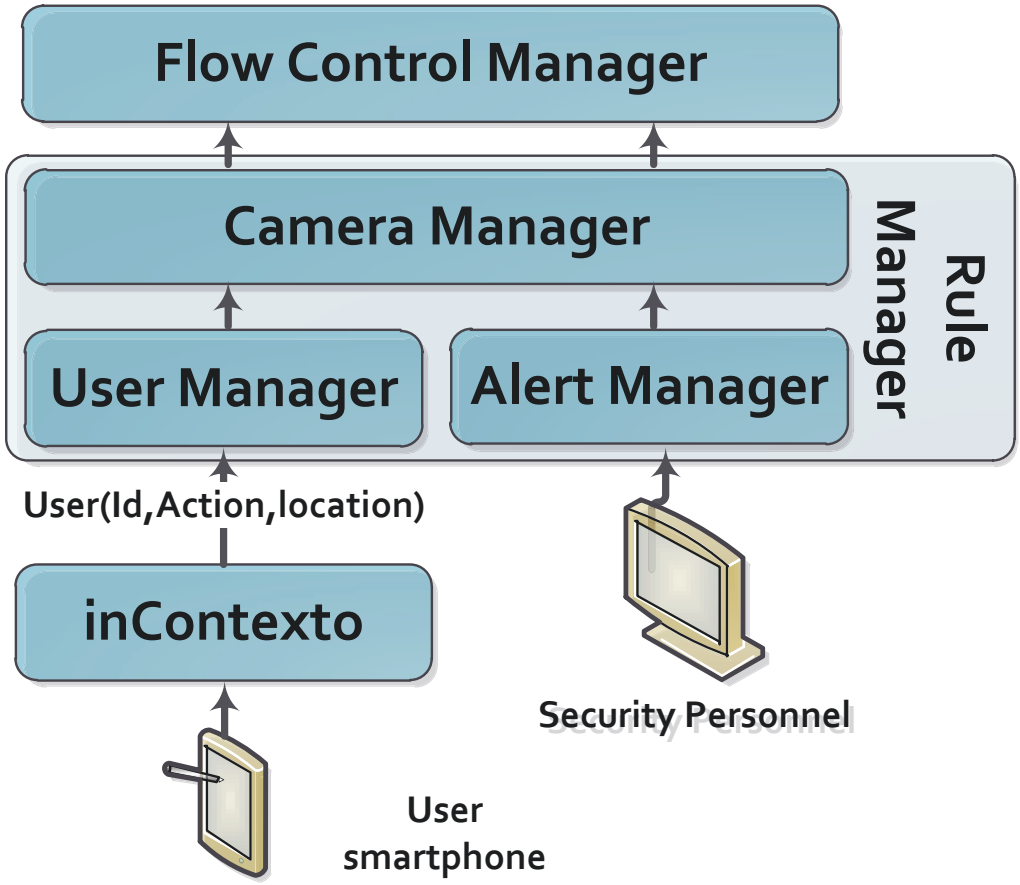

Figure 4.6: ContextCare architecture: Rule Manager Component, video surveillance system and inContexto.

that provides background processing. Background processing provides us in a non-intrusive way to obtain patient context. As it was explained before this level gathers single raw data from smartphone sensors (Accelerometer, Gyroscope, GPS and a heart rate band connected via Bluetooth) in order process and transform in features. Patient context data (activity, fall detection and heart rate monitor) is accessed through Android OS API, specifically sensor manager class that provides methods to obtain all the mobile sensors. In order to infer activity and people falling inContexto collects the following information Signal mean and variance, correlation between axes and signal energy, as it was explained in Chapter 3.

Activity recognition module uses selected features to infer what activity is the user engaged in. As well as level 0, Context Care level 1 is implemented following the guidelines depicted in Chapter 3. However, it is included a fall detection tracking using accelerometer values (Kazi et al., 2014). ContextCare level 2 just collect information from security personnel in order to

decide which one is the most suitable to attend the warning.

Finally, high-level action reasoning level (inContexto level 3) aims to compose all the received action from the activity recognition level into a Global action for each user. Beyond the standard reasoning model based on the ontology mechanism, it is possible to perform rule based inferences using a description logic inference engine. At the beginning, this rule would be described by an expert in order to teach the system.

4.4.3 Video Surveillance architecture

The Video Surveillance architecture is basically a system that allows the control of PTZ cameras by local or remote processes. In this case, this architecture mainly resides in the *Sensor Manager*, which is the responsible of attending control flows (allowing the positioning of PTZ cameras), and streaming video sequences to the different terminals. ContextCare uses this component for monitoring patients, allowing an easy management of the underlying cameras. The *Sensor Manager* used by ContextCare is the responsible of video acquisition, compression, and transmission, as well as to handle the communication protocols to perform the different movements in the cameras. The internal organization of this component is briefly outlined in figure 4.7.

There are two main functions that are controlled by the *Sensor Manager* for each video camera. The first one is related with the control of the camera and its movements. As we can see in figure 4.7, there is a PTZ server which allows other processes to interact with the camera, so this controller can provide a standard interface to control homogeneously any underlying device. It has defined some high-level operating primitives like *goTo X Y*, *zoom amount*, etc. These high-level primitives are exposed as a non-connection oriented UDP Server, with a simple request-response protocol in the client-server computing mode.

The second main feature is about video acquisition and transmission. Generally, the access to limited resources like video devices is a problem if share the information between different terminals or systems is a requirement. In order to solve this issue we have defined two different strategies depending on the video destination. For local processes running on the same computer it is created a shared region of non-paged memory, which can be accessed to retrieve the latest video frames.

On the other hand, for remote processes,a JPEG2000 compression (Luis and Patricio, 2009) and real-time streaming system (Bustamante et al., 2011) is used with the aim of provide frames with the minimum delay. This allows transmitting real-time video sequences to remote processes like operators, agents, backup systems, mobile phones, etc.

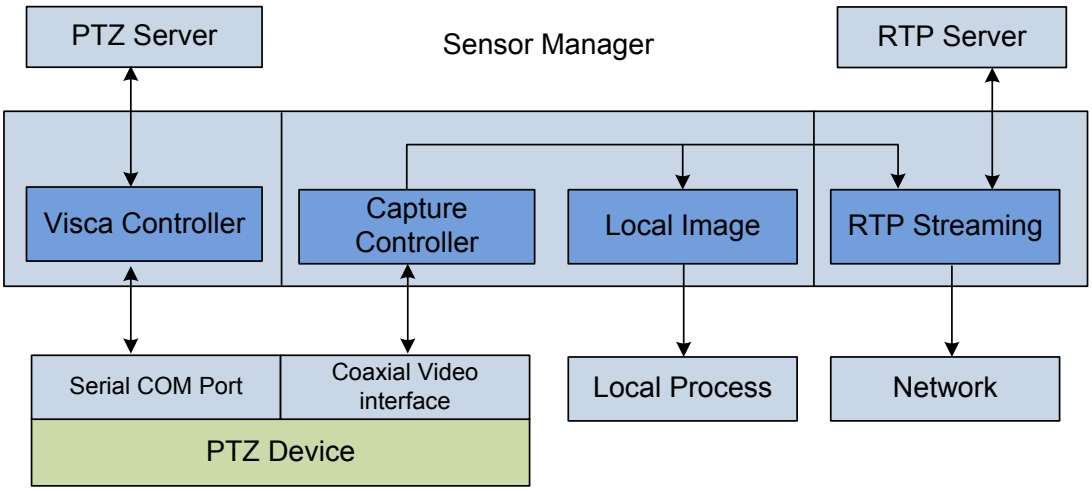

Figure 4.7: Sensor manager for local/remote camera control

It is implemented under a Real-time Transport Protocol (RTP) server (RTP Server on figure 4.7) based on JPEG2000 image sequences, as described in RFC5371 (for Standardization et al., 2000). This standard defines a new RTP extension, which allows to transmit JPEG2000 frames over RTP packets. Moreover, this implementation of the standard introduces a real-time motion compensation technique, which is still patent pending (Spanish patent number P200900260) and still complaints with RFC5371.

4.4.4 inContexto level 3: Rule Manager Component

Rule Manager (See figure 4.6) component is probably the most important component of the ContextCare architecture. it involves many IT disciplines like database access, knowledge reasoning, video camera resource management and ECA rules evaluator. ECA model allows bidirectional communication between users (patients and security personnel) and video surveillance system in order to make possible to control and coordinate human resources and video cameras.

Rule Manager is a centralized server where the whole surveillance system is managed. Rule manager component's inputs are ECA rules provided by surveillance personnel which has the form: ON *event* IF *condition(s)* DO *action(s)*. Besides, user context activity information from inContexto architecture which follows this structure: *(userID, action, location)*. On the contrary, rule manager outputs are guidelines (ECA rules actions) to carry out by the flow control component, coordinating cameras and also inform to the surveillance staff that something wrong is happening.

When ContextCare receives a new user context action from inContexto architecture, ECA

rules is constantly evaluated in order to detect configured events, executing the associated actions if the conditions are fulfilled. In that case, first of all a message sequence is activated to notify from the users to Security personnel that something is happening. Later, ContextCare architecture decides which camera is the most propitious to track the situation. Finally, the system starts to monitor autonomously the involved person.

User Manager receives and stores every user context action and location generated by inContexto architecture. This component manages inContexto User State Message (USM) and also stores into a database with the purpose of providing this information to camera manager, which will check if any alert is active. inContexto USM is composed by the following attributes:

- *User id* which consist in a string which identifies every user, in this case smartphone direction IP. The Id permits to determines if the performed action is allowed to this person or not.
- *Action*, this field contains a concrete action or sensor value (depending on the embedded sensors on the system). Smartphone normally provides inertial sensors, however, inContexto allows to connect other sensors such as blood pressure, hearting beat sensor, and so on.
- *Location* this field depicts the place where the user is in that moment. The suitability of each method depends on whether the location is outdoors or indoors and also the technology used. This field necessarily contains an absolute position like GPS coordinates. It could be filled by a symbolic position like corridor, room number, the nearest access point or wherever.

Alert manager aims to communicate ContextCare architecture with human resources staff (Security or emergency personnel) terminals. The set of ECA rules are configured by an expert and define those events a flow control manager should be aware of. As we explain previously, three fields compose the ECA Rules (events, conditions, actions) and they are described below:

- *Events*: describes a situation (user activity or location in this case) to which the rule may be able to respond. Events can be essentially divided into two categories: (i) primitive events, which correspond to elementary occurrences, and (ii) composite events that are composed for more than one primitive events.
- *Conditions*: specifies the conditions to trigger the ECA rule. Once the result of the condition evaluation is true, the condition is satisfied and the action field is executed.

- *Actions*: describes the task the rule considers relevant to the event and the condition. Actions field indicates the subsequent activities if the condition is satisfied.

Alert manager component generates two different responses, the first one is ECA rules created from the human resources terminal to Camera Manager and the second one is the alert protocol when a rule is triggered in the Camera Manager Component. Camera Manager component contains ECA Rules engine which is responsible of generate alerts and guidelines to manage the video surveillance system. These rules involves to evaluate 'online' conditions, i.e., those which require to access an external resource (user context information, video camera control), therefore it is mandatory a real time response.

Mainly, there are two ways to control video flow. The first one is manually via the main terminal. Human Resources moves manually the cameras looking for situation of interest. The second way to control video flow is automatically via actions defined by ECA rules manager. Hence, it operates video cameras three different actors but each one play different roles:

- *Human Resources* monitoring the video streamed by different cameras and controlling their orientation manually.
- *Mobile human resources* walking around the monitored area. This person has a smartphone where receives alerts with security problems.
- *Users* are the principal actor. They also have a smartphone, which is used to track their actions. When a non-usual action takes place the mobile phone will launch an alert to Rule manager.

Camera Manager gathers user context actions and ECA rules to build a unified view of the scene. Besides, it creates a goal, which represent the overall objective of the video surveillance system. For example, a goal may be *track user which ID is 10001* and Camera Manager send user information (location and action) to the control flow manager which will be decided which camera is the most suitable to track this person. When an rule is triggered, Camera Manager informs user position to Control flow component in order to track him/her.

Finally, Camera manager creates a User State Message and sends it to emergency services in order to solve the problem soon. Summarizing, Camera Manager decides according to the environment situation and the ECA rules which goal is a priority, sorting a list of goals actions for the Control Flow Manager and creating the alert to the Security personnel.

4.4.5 ContextCare Architecture Evaluation

ContextCare architecture has been created to monitor autonomously dependent people (Elderly people, children, etc.) and give health care support to hospital patients. Taking into account that this architecture is an autonomous monitor system, depending on the ECA Rules depicted by the human resources, it is mandatory a real-time performance to efficiently manage the information given by sensors.

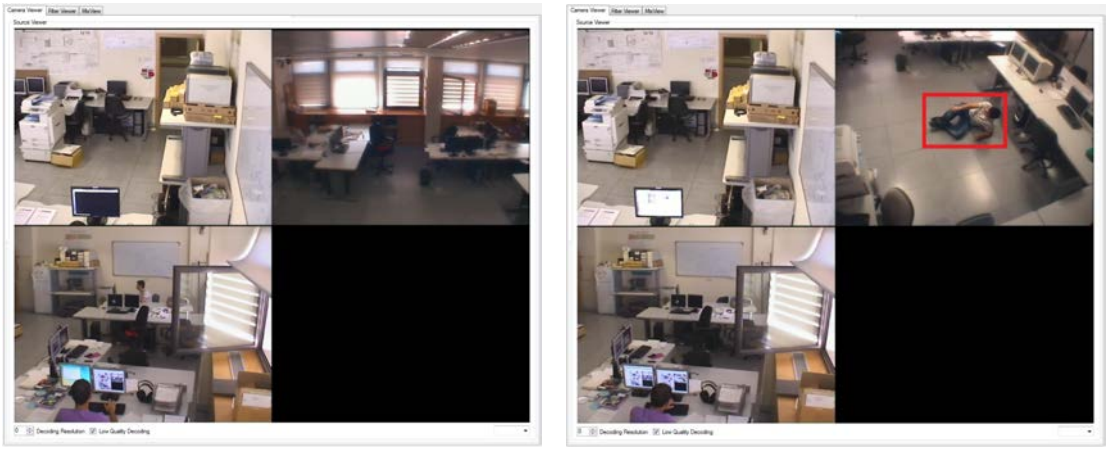

Figure 4.8: Video surveillance systems shows a person who fell down on the floor according to inContexto results. The first figure shows the monitored environment and th second one shows a screen alert over the person.

In order to better assist the evaluation process the ContextCare system deployment will take place firstly in the University Carlos III of Madrid for pre-evaluation. The working scenario consists in a visual sensor network, which contains six Pant-Tilt-Zoom (PTZ) control video devices, around two rooms, three in the corridor and the other ones in the main room.

The scenario is a public place, so every person who enter to the track zone was under system control, however, there are just four people (playing user role) connected to ContextCare Application. In this case, inContexto was configured to recognize five different actions (Lying, Standing, Walking, Running and Falling down).

In this case, an expert defines the ECA rules, which describes what is considered a problem for each patient. Thereby, the ContextCare system may reach different rules like *'ON userContex:fall_down IF user:X DO follow'* or *'ON userContex:walking IF user=Y DO track'*). Every single action is detected by inContexto architecture and send it to ContextCare application where it is checked in the rule-based engine if it actives a rule.

First figure 4.8 depicts video surveillance system tracking. There are two persons sitting inside the room (bottom left camera), one of them stands up and start to walk. inContexto

generates two different USM as follows:

- *(userID:10001,action:Stand,location:corridor)*
- *(userID:10001,action:Walking,location:(2,1))*

First alert shows a symbolic location *corridor* and the second one an absolute position according to video surveillance coordinates. After that, the user fall down and the smartphone generates another USM, which dispatches the next rule in the camera Manager Component. Alert manager rule engine checks the actual situation and the next ECA rule is dispatched.

- *'ON userContex:fall_down IF user:10001 DO follow AND Inform SP'*

Finally, alert system inform to camera manager component where is the user (Figure 4.8 shows the person on the floor inside a red rectangle) and also it creates the message to inform every security personnel.

4.5 Summary and Conclusion

The most critical part of this work has focused on developing a framework to infer and evolve people context information from multiple sources. The proposed architecture distributes the processing load between mobile device and a server placed on the cloud. The first part of the chapter presents the development of inContexto architecture. The architecture is split in two different modules, the first one is on the server side where the API REST is implemented and the second one is on the client side where an Android mobile application was develop in order to collect user context information.

Subsequently, a study case is described in order to evaluate the proposal architecture focus on a business management scenario. AmI technology is developing fast and will promote some characteristics in the area of context awareness, anticipatory behaviour and video surveillance. ContextCare improves multicamera tracking applications performance using information from smartphones in a eHealth care scenario. Selecting the most suitable camera for any situation assessment during video tracking analysis. Using this new approach, the time the security personnel spends in front of the screen is reduced, taking this time for other tasks.

inContexto architecture and its special implementation in ContextCare is well-suited in many human resources surveillance situations, for example the prevention of labour risks, management of human resources. However, we think that eHealth context is the most suitable situation.

Considered future works extending the development of the Activity Recognition system with more complex activities is to create an application, which is able to send a sms or call to your relatives.

Conclusions

C.1 Final Remarks

NOWADAYS, AmI technology is developing fast and will promote a new generation on business management including the area of context awareness, anticipatory behavior and video surveillance. Although there are already solutions that have been successfully implemented to infer people context, there are several important issues not tackled in these kinds of approaches yes. For example, non-intrusive sensors, a generic architecture, etc. Some of this contributions are directly benefiting the research community. Moreover, this thesis also deals with the implementation of the presented methods, in order to accomplish the envisioned mobile system for physical and emotional activity monitoring.

The main goal stated for this thesis was the development of a framework which collects and infers people context based only on data obtained using wearable sensors and Social Networks Sites (SNS). Besides, this architecture aims to ease people everyday tasks. We introduced the design, implementation, evaluation, and user experiences of the inContexto framework which represents a system that combines the inference of people sensing presence. Social Networks Sites and smartphone in-built sensors are becoming a novel proposal in user context information systems. These two new sensors provide great amount of user information in an unobtrusive way.

Supported by advanced sensing capabilities and increasing computational resources, smartphones will become our virtual companions; able to learn our daily routines, react, and propose solutions tailored to personal behaviors and habits. In this thesis we have taken steps towards the realization of this new personal and wearable sensors, by proposing applications, frameworks, and algorithmic solutions.

The goal was motivated by the fact that inferring people context will help us in everyday tasks. Due to highly change of the people context information, the software system running on them has to face problems such as: user mobility and availability, resources or goal changes, which may happen in any moment. To cope with these problems, such system must senses the environment and the user and acts on it, over time in pursuit of its own benefit.

Therefore, another central spotlight of this thesis was to explain how user context information flows from wearable sensors to high-level context information. Emphasis was placed thereby on identifying key challenges in this research field and on addressing them with the introduction of novel methods and algorithms.

Two secondary objectives were also defined in this context, on the one hand to create a set of basic physical actions and on the other hand to define a set of emotional states. This goal allow us to improve the current user context definition: To distinguish activities of and basic postures (run, walk, lay and drive), and emotional states (fear, anger, sadness, happiness, surprise, and disgust). Moreover, it should be noted that a high value was put on the evaluation of the proposed methods: Thorough experiments are presented in the respective chapters of this thesis to justify the introduced data processing and classification methods.

Finally, another direction to take into account is the integration of the developed mobile system into a full health-care application for aerobic activity monitoring and support in daily life decisions. The listed contributions are both of theoretical (cf. e.g. the novel algorithms and developed models) and of practical value (cf. e.g. the proposed evaluation techniques).

C.2 Future work

There are a few different areas that should be investigated further in terms of this thesis. First of all, standardization scheme of the communication protocol in order to more accurately describe certain types of systems. Currently, these types can be any arbitrary name, however there should be a standard naming convention for the sensor types and every inContexto implemented techniques (extracted features, machine learning technique, etc.). This naming convention could be implemented following the guidelines of SensorWeb Enablement (SWE) family of standards.

Also, there are changes that need to be made to the architecture in order to scale the platform as a whole. As more sensors and users are using the platform and query requirements diversify, scalability becomes an issue in the inContexto framework. Managing a large amount of information with one registry is not feasible without having performance issues related to queries and management. NoSql databases or big data approach may be a simple solution to take into account.

Furthermore, there are some issues related to including new inferring methods that need to be addressed. As we explained before the interface naming standard should be faced in order to select the best technique automatically depending on the current features collected. Besides,

to integrate more sophisticated analysis services that can be provided as clients or systems by using the capabilities of popular statistical and signal processing software components such as R, Octave, and Matlab.

Finally, some additional systems and clients need to be made for the architecture that can act as services that other components could use. Nevertheless, inContexto API offers to third party applications a standard way to generate new services.

A

Published Results

THIS appendix lists the scientific works that have to do with this thesis organized by class of publication.

Articles

- Gil, G. B., Berlanga, A., and Molina, J. M. (2009). Estudio del futuro de las tecnologías inalámbricas de comunicaciones en la seguridad del tráfico ferroviario. *Estudios de construcción y transportes*, (111):91–114

- Gil, G. B., Berlanga, A., and Molina, J. M. (2012c). Incontexto: multisensor architecture to obtain people context from smartphones. *International Journal of Distributed Sensor Networks*, 2012

Conferences Publications

- Gil, G. B., de Jesús, A. B., and Lopéz, J. M. M. (2010). Multi-sensor and multi agents architecture for indoor location. In *Distributed Computing and Artificial Intelligence*, pages 309–316. Springer

- Gil, G., Berlanga de Jesus, A., and Molina Lopez, J. (2011a). incontexto: A fusion architecture to obtain mobile context. In *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*, pages 1–8. IEEE

- Gil, G. B., Berlanga, A., and Molina, J. M. (2011b). Physical actions architecture: Context-aware activity recognition in mobile devices. In *User-Centric Technologies and Applications*, pages 19–27. Springer

- Gil, G. B., Berlanga, A., and Molina, J. M. (2012b). Emotioncontext: user emotion dataset using smartphones. In *Ambient Assisted Living and Home Care*, pages 371–374. Springer

- Gil, G. B., Bustamante, A. L., Berlanga, A., and Molina, J. M. (2012d). Contextcare: Autonomous video surveillance system using multi-camera and smartphones. *Management Intelligent Systems*, pages 47–56

- Gil, G., de Jesús, A., and Lopéz, J. (2012a). Comparing features extraction techniques using j48 for activity recognition on mobile phones. *Distributed Computing and Artificial Intelligence*, pages 141–150

- Gil, G. B., de Jesús, A. B., and Lopéz, J. M. M. (2013). Combining machine learning techniques and natural language processing to infer emotions using spanish twitter corpus. In *Highlights on Practical Applications of Agents and Multi-Agent Systems*, pages 149–157. Springer

B

EmotionContext: User Emotion Dataset Using Smartphones

RECENTLY, increasing attention has been directed to the study of the human emotional state. Affective Computing (AC) or Emotion-oriented computing is a branch of AI that deals with the design of systems and devices that can recognize, interpret, and process human affective states. AC term was introduced by Rosalind Picard at MIT in 1997 (Picard, 1997). Pichard described three types of affective computing applications: 1) systems which detect user emotions, 2) systems that express what a human would perceive as an emotion (e.g., an avatar, robot, and animated conversational agent), and 3) systems that can actually "feel" an emotion.

AC is traditionally carried out through video systems (Reilly et al., 2008) or through intrusive systems which make difficult to implement in real applications due to user's reluctance to wear devices across their bodies. Smartphones are particularly well-suited to accomplish this task. They are considered a wearable system and also they can operate during long periods of time sensing user activities, routines and the environment.

However, recently researches are engaging in solve affective computing problem using smartphones. Moreover, smartphones are changing communication channels computer mediated communication (CMC). Typically, e-mail, social networks (asynchronous) sms, or even instant messaging (synchronous). New communication channels reduce personal contact between speakers to zero. For example, text messaging and social networks are the most popular way to communicate for teenagers.

They are provided by countless number of sensors such as microphones and digital cameras and recently they have been equipped with new sensors (accelerometer, gyroscope, digital compass, proximity sensor, light sensor, GPS, etc) (Blazquez Gil et al., 2011). Taking the

advantage of these new features, it is possible to create an application which gather user information in order to infer user emotional state.

This appendix presents the most widely techniques and methods in the literature to infer human emotional state (Facial Expression, Emotional Speech, Body Gesture, Contextual information and Multi-Sensor solutions) and which techniques are able to implement on a smartphones. Finally, it presents ContextCare, an architecture to obtain labeled emotion information from smartphones.

B.1 EmotionContext dataset

Nowadays, smartphones are not just a tool for communication (telephone or internet), besides, they have embedded sensors which provide countless information about the user. Hence, smartphones can (at least theoretically) hear what you hear, see what you see and even read what you read.

Sensor	API Class	Techniques	Emotions
Front Camera	MediaRecorder	Facial Expression	6 Basic & neutral
Microphone	MediaRecorder	Emotional Speech	6 Basic & neutral
Accelerometer	SensorManager	Hand movements	Happiness, anger & Neutral
Gyroscope		&	
Location	LocationManager	-	Happiness & Neutral
Typing	Sensing Module	Frequency	Anger & Neutral
SN status	Facebook & Twitter	NLP	6 Basic & neutral

Table B.1: EmotionContext Dataset Proposal: Matching between smartphone sensor and emotion recognition technique.

Although camera and microphone are the most used sensors in emotions recognition systems. Facial and speech emotion recognition techniques are computational costly even using smartphones. Table 2.3 shows what sensors could be used to obtain emotion information. Moreover, thanks to the wireless smartphone connection (Bluetooth or Wifi) is possible to connect new sensors such as heart beat sensors, pressure sensors and so on.

One of the main goals in smartphone application community research is to recognize inconspicuously activity (physical or mental) of individuals and react to their needs. Taking advantage of these features researchers have been used these new data in order to obtain physical user improving user smartphone experience (Blazquez Gil et al., 2011).

Smartphone has been chosen by the big amount of data that provides and also because it is well-suited to obtain this data in an non-intrusive way. Other possible sensor to provide emotion

information is accelerometer and gyroscope. If the user is moving unconsciously her/his hands, it may be possible to detect a emotional state and infer is the user is nervous or not. User location does not provide a direct response to human emotions but it provides more specific information about the user. According to user position is possible to infer the user is having fun or not (e.g. he/she is dancing in a disco or he/she is at work).

- A-GPS: which improves the performance by adding information, through another data connection (Internet or other), minimizing the amount of time and information is required from the satellites. Smartphones GPS precision is about 2-3 meters.
- GSM cell tower triangulation: Energy consumption is less than GPS. On the contrary the precision is also lower, about 30 meters.
- Internet connection: Using Geolocation API [1] is possible to locate a client device using Wifi. Although Geolocation API provides latitude and longitude position, the real power consist is that provides logical location.

Finally, in (Blazquez Gil et al., 2011) was introduced Social Networks Sites (Facebook, Twitter, etc.) as user context sensor. Using Natural processing languages techniques is possible obtain user emotions from user SNS status.

B.1.1 EmotionContext architecture

The architecture of our framework is depicted in figure B.1. The schema shows the three modules: Smartphone app, SN server and the EmotionContext backend server.

Firstly, SN API's enable developers to access some of the core primitives of each SN including timelines, status updates, pictures, etc. This information is reachable through OAuth authentication protocol which grants access to user information to third-party applications. When users introduce their Facebook or Twitter credentials, they give back an access token, a string denoting a specific scope, lifetime, and other access attributes.

Secondly, EmotionContext smartphone app is developed in Android OS and contains three different modules: Sensor, pre-processing and communication module. Sensor module is an Android service which continuously gathers smartphone sensor information according to table 2.3. Accelerometer and gyroscope (MEMS) sampling frequency is not constant in Android OS devices, depends on the mobile phone features, OS version and computational workload but it

[1] Geolocation API http://dev.w3.org/geo/api/spec-source.html

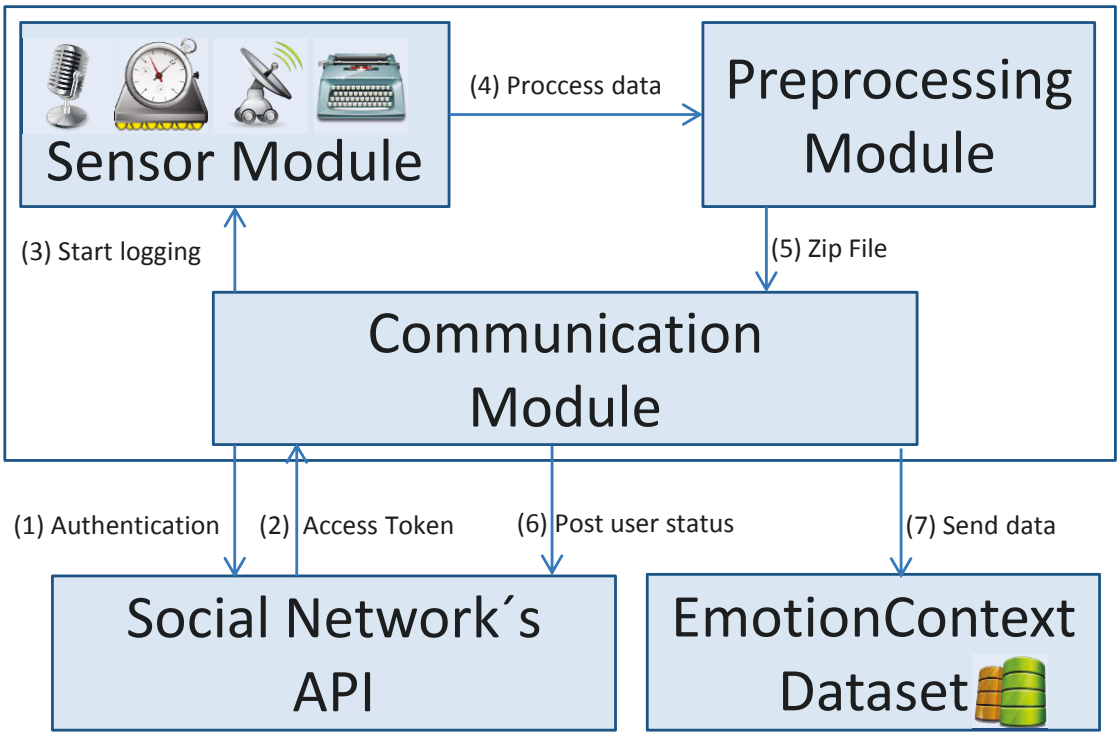

Figure B.1: EmotionContext requests authorization from the SN API using OAuth using HTTP Protocol (1). Below, OAuth Authorization serves sends an access token which grants EmotionContext access to user SN protected resources (2). At the same time, Sensor module start to log smartphone data (3). When the user finish writing her/his post, sensor module send all the information to the preprocessing module (4) which select the best samples and compress them (5). Finally, Communication Module sends User personal status to SN (6) and also it sends .zip file to EmotionContext dataset via PHP (7).

was chosen the fastest one. Microphone and front camera data is stored following the Google API audio and Video codecs (Video 2 Mbps frame rate and 192 Kbps audio bitrate). In order to reduce signal noise and feature extraction problems, the maximum quality codec has been chosen in both cases.

The preprocessing module transforms raw data to a vector features. MEMS data have been stored into a sliding window of 512 samples (approximately 5 seconds), 256 of which overlap with consecutive ones. Besides, the first and last 10% of windows stored are cropped in order to avoid outliners data, and finally stored in a plain text file. As well as MEMS sensors, audio and video data have been cropped. All the files are compressed in a .zip file which makes easier to send it to a server using PHP.

Every vector features is composed by a 3 plain text with MEMS information, 1 video file from the smartphone front camera, 1 audio file with the recorded audio and the user SN status. Finally, the user labels every vector according their emotional state. Communication Module connects via PHP, Emotion Context Smartphone app, SN API's and EmotionContext Dataset.

EmotionContext Dataset server is a LAMP distribution (Linux, Apache, MySQL and PHP/Python). LAMP distribution support large file size (greater than 2 GB), bandwidth throttling to limit the speed of responses in order to not saturate the network and also provides Server-side scripting (PHP) to store every vector features in the MySql database.

Figure B.2: ContextCare mobile phone main screen. Users could post their information on Social Network Sites.

Finally, figures B.2 and B.3 show EmotionContext Andorid app screen flow. The first one shows sensors and it finish when the user press the post button. Since EmotionContext is a supervised system, the user must to decide which emotion is feeling. Finally, EmotionContext ask to the user which emotional state is. User may choose between the six basics emotions or Neutral one.

- Video Codec: H.264 AVC, 1280 x 720 px resolution, and 2 Mbps frame rate.
- Audio Codec: AAC-LC and 192 Kbps audio bitrate.

Finally, GPS, typing frequency, text and labelled emotion is stored when the user post their

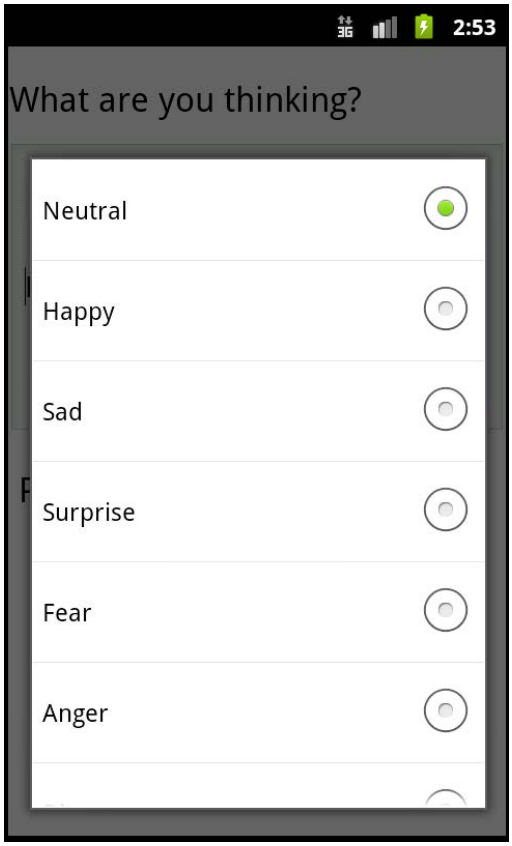

Figure B.3: Once the users have written a comment, thy can select the actual emotional state in order to tag the comment.

state since is a discrete data. GPS may be considered to be tracked but it is not clear the real utility of the features (speed, various position) from this sensor.

B.2 Conclusion

This work shows the most relevant works in emotion recognition area and also it proposes a smartphone dataset (EmotionContext) which combines information from a smartphone embedded sensor. EmotionContext dataset is proposed to satisfy a new wave in smartphones application research community. This section describes pros and cons to use smartphones sensors in order to obtain user emotion.

It is important to highlight that most of the proposed architecture to create the dataset is highly reusable in future apps. Sensing and preprocessing module will be nearly the same in an app able to obtain user emotions. Considering future works are to make a study of the extracted data from the smartphones and of course make an application able to discern the

human emotional state. This kind of applications will be very useful in eHealth application, for example monitoring people with mental diseases and sending an alert when the patient is having a disorder.

References

Abowd, G., Atkeson, C., Hong, J., Long, S., Kooper, R., and Pinkerton, M. (1997). Cyberguide: A mobile context-aware tour guide. *Wireless networks*, 3(5):421–433.

Alimoglu, F. and Alpaydin, E. (1996). Methods of combining multiple classifiers based on different representations for pen-based handwritten digit recognition. In *Proceedings of the Fifth Turkish Artificial Intelligence and Artificial Neural Networks Symposium (TAINN 96*. Citeseer.

Anderson, K. and McOwan, P. (2006). A real-time automated system for the recognition of human facial expressions. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 36(1):96–105.

Avci, A., Bosch, S., Marin-Perianu, M., Marin-Perianu, R., and Havinga, P. (2010). Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey. In *Architecture of Computing Systems (ARCS), 2010 23rd International Conference on*, pages 1–10. VDE.

Bao, L. and Intille, S. (2004). Activity recognition from user-annotated acceleration data. *Pervasive Computing*, pages 1–17.

Barralon, P., Noury, N., and Vuillerme, N. (2006a). Classification of daily physical activities from a single kinematic sensor. In *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, pages 2447–2450. IEEE.

Barralon, P., Vuillerme, N., and Noury, N. (2006b). Walk detection with a kinematic sensor: frequency and wavelet comparison. *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, 1:1711–4.

Begole, J. B., Tang, J. C., and Hill, R. (2003). Rhythm modeling, visualizations and applications. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 11–20. ACM.

Bianchi-Berthouze, N. and Kleinsmith, A. (2003). A categorical approach to affective gesture recognition. *Connection Science*, 15(4):259–269.

Bidargaddi, N., Sarela, A., Klingbeil, L., and Karunanithi, M. (2007). Detecting walking activity in cardiac rehabilitation by using accelerometer. In *Intelligent Sensors, Sensor Networks and Information, 2007. ISSNIP 2007. 3rd International Conference on*, pages 555–560. IEEE.

Blazquez Gil, G., Berlanga, A., and Molina, J. M. (2011). inContexto: A fusion architecture to obtain mobile context. In *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*, pages 1–8. IEEE.

Boyd, D. and Ellison, N. (2008). Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230.

Bustamante, A., Molina, J., and Patricio, M. (2011). Multi-camera control and video transmission architecture for distributed systems. *User-Centric Technologies and Applications*, pages 37–45.

Cairns, D. and Hansen, J. (1994). Nonlinear analysis and classification of speech under stressed. *J. Acoust. Soc. Am*, 96(6).

Castellano, G., Villalba, S., and Camurri, A. (2007). Recognising human emotions from body movement and gesture dynamics. *Affective computing and intelligent interaction*, pages 71–82.

Chen, C., Anton, S., and Helal, A. (2008). A brief survey of physical activity monitoring devices. *University of Florida Tech Report MPCL-08-09*.

Chen, H. (2004). *An Intelligent Broker Architecture for Pervasive Context-Aware Systems*. PhD thesis, University of Maryland, Baltimore County.

Chon, J. (2011). LifeMap: Smartphone-based Context Provider for Location-based Services. *IEEE Pervasive Computing*, pages 1–7.

Chon, J. and Cha, H. (2011). LifeMap: Smartphone-based Context Provider for Location-based Services. *IEEE Pervasive Computing*.

Cilla, R., Patricio, M., Berlanga, A., and Molina, J. (2011). A probabilistic, discriminative and distributed system for the recognition of human actions from multiple views. *Neurocomputing*.

Clarke, J., Lethbridge, J., Liu, R. P., and Terhorst, A. (2009). Integrating mobile telephone based sensor networks into the sensor web. In *Sensors, 2009 IEEE*, pages 1010–1014. IEEE.

Clarkson, B. and Pentland, A. (1999). Unsupervised clustering of ambulatory audio and video. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 6, pages 3037–3040. IEEE.

Cleland, I., Kikhia, B., Nugent, C., Boytsov, A., Hallberg, J., Synnes, K., McClean, S., and Finlay, D. (2013). Optimal placement of accelerometers for the detection of everyday activities. *Sensors*, 13(7):9183–9200.

Consolvo, S., McDonald, D., Toscos, T., Chen, M., Froehlich, J., Harrison, B., Klasnja, P., LaMarca, A., LeGrand, L., Libby, R., et al. (2008). Activity sensing in the wild: a field trial of ubifit garden. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1797–1806. ACM.

Cook, D., Augusto, J., and Jakkula, V. (2009). Ambient intelligence: Technologies, applications, and opportunities. *Pervasive and Mobile Computing*, 5(4):277–298.

Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., and Taylor, J. (2001). Emotion recognition in human-computer interaction. *Signal Processing Magazine, IEEE*, 18(1):32–80.

Crouter, S., Schneider, P., Karabulut, M., and Bassett, D. (2003). Validity of 10 electronic pedometers for measuring steps, distance, and energy cost. *Medicine and Science in Sports and Exercise*, 35(8):1455–1460.

Darwin, C., Ekman, P., and Prodger, P. (2002). *The expression of the emotions in man and animals*. Oxford University Press, USA.

De Choudhury, M., Counts, S., and Gamon, M. (2012). Not all moods are created equal! exploring human emotional states in social media. In *Sixth International AAAI Conference on Weblogs and Social Media*.

De Silva, L. and Ng, P. (2000). Bimodal emotion recognition. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 332–335. IEEE.

Derks, D., Bos, A., and Von Grumbkow, J. (2008). Emoticons and online message interpretation. *Social Science Computer Review*, 26(3):379–388.

DeVaul, R. and Dunn, S. (2001). Real-time motion classification for wearable computing applications.

Dey, A. (2000). *Providing architectural support for building context-aware applications*. PhD thesis, Georgia Institute of Technology.

Dey, A. and Abowd, G. (2000). Towards a better understanding of context and context-awareness. In *CHI 2000 workshop on the what, who, where, when, and how of context-awareness*, volume 4, pages 1–6. Citeseer.

Eagle, N. and Pentland, A. (2006). Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268.

Ekman, P. and Friesen, W. (1978). Facial action coding system: A technique for the measurement of facial movement.

El Ayadi, M., Kamel, M., and Karray, F. (2011). Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern Recognition*, 44(3):572–587.

Ellison, N. et al. (2007). Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230.

Ermes, M., Parkka, J., and Cluitmans, L. (2008). Advancing from offline to online activity recognition with wearable sensors. In *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pages 4451–4454. IEEE.

Essa, I. and Pentland, A. (1997). Coding, analysis, interpretation, and recognition of facial expressions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):757–763.

Esteban, J., Starr, A., Willetts, R., Hannah, P., and Bryanston-Cross, P. (2005). A review of data fusion models and architectures: towards engineering guidelines. *Neural Computing & Applications*, 14(4):273–281.

Fasel, B. and Luettin, J. (2003). Automatic facial expression analysis: a survey. *Pattern Recognition*, 36(1):259–275.

Fattoruso, G., Tebano, C., Agresta, A., Buonanno, A., De Rosa, L., De Vito, S., and Di Francia, G. (2015). Applying the swe framework in smart water utilities domain. In *Sensors*, pages 321–325. Springer.

Fielding, R. (2000). *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. (2009). Rfc 2616: Hypertext transfer protocol–http/1.1, 1999. *URL http://www. rfc. net/rfc2616. html*.

Fielding, R. T. and Taylor, R. N. (2002). Principled design of the modern web architecture. *ACM Transactions on Internet Technology (TOIT)*, 2(2):115–150.

for Standardization, I. O., Commission, I. E., et al. (2000). Iso/iec 15444-1: 2000. *Information technology JPEG 2000 image coding system Part 1: Core coding system*.

Galeana-Zapién, H., Torres-Huitzil, C., and Rubio-Loyola, J. (2014). Mobile phone middleware architecture for energy and context awareness in location-based services. *Sensors*, 14(12):23673–23696.

Gil, G., Berlanga de Jesus, A., and Molina Lopez, J. (2011a). incontexto: A fusion architecture to obtain mobile context. In *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*, pages 1–8. IEEE.

Gil, G., de Jesús, A., and Lopéz, J. (2012a). Comparing features extraction techniques using j48 for activity recognition on mobile phones. *Distributed Computing and Artificial Intelligence*, pages 141–150.

Gil, G. B., Berlanga, A., and Molina, J. M. (2009). Estudio del futuro de las tecnologías inalámbricas de comunicaciones en la seguridad del tráfico ferroviario. *Estudios de construcción y transportes*, (111):91–114.

Gil, G. B., Berlanga, A., and Molina, J. M. (2011b). Physical actions architecture: Context-aware activity recognition in mobile devices. In *User-Centric Technologies and Applications*, pages 19–27. Springer.

Gil, G. B., Berlanga, A., and Molina, J. M. (2012b). Emotioncontext: user emotion dataset using smartphones. In *Ambient Assisted Living and Home Care*, pages 371–374. Springer.

Gil, G. B., Berlanga, A., and Molina, J. M. (2012c). Incontexto: multisensor architecture to obtain people context from smartphones. *International Journal of Distributed Sensor Networks*, 2012.

Gil, G. B., Bustamante, A. L., Berlanga, A., and Molina, J. M. (2012d). Contextcare: Autonomous video surveillance system using multi-camera and smartphones. *Management Intelligent Systems*, pages 47–56.

Gil, G. B., de Jesús, A. B., and Lopéz, J. M. M. (2010). Multi-sensor and multi agents architecture for indoor location. In *Distributed Computing and Artificial Intelligence*, pages 309–316. Springer.

Gil, G. B., de Jesús, A. B., and Lopéz, J. M. M. (2013). Combining machine learning techniques and natural language processing to infer emotions using spanish twitter corpus. In *Highlights on Practical Applications of Agents and Multi-Agent Systems*, pages 149–157. Springer.

Goldwasser, S., Micali, S., and Rackoff, C. (1989). The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208.

Gómez-Romero, J., Serrano, M., Patricio, M., García, J., and Molina, J. (2011). Context-based scene recognition from visual data in smart homes: an information fusion approach. *Personal and Ubiquitous Computing*, pages 1–23.

Gonzalez, M. C., Hidalgo, C. A., and Barabasi, A.-L. (2008). Understanding individual human mobility patterns. *Nature*, 453(7196):779–782.

Gordon, D., Czerny, J., Miyaki, T., and Beigl, M. (2012). Energy-efficient activity recognition using prediction. In *Wearable Computers (ISWC), 2012 16th International Symposium on*, pages 29–36. IEEE.

Grimm, M., Kroschel, K., Mower, E., and Narayanan, S. (2007). Primitives-based evaluation and estimation of emotions in speech. *Speech Communication*, 49(10-11):787–800.

Gross, T. and Specht, M. (2001). Awareness in context-aware information systems. In *Mensch & Computer 2001*, pages 173–182. Springer.

Guiry, J. J., van de Ven, P., Nelson, J., Warmerdam, L., and Riper, H. (2014). Activity recognition with smartphone support. *Medical engineering & physics*, 36(6):670–675.

Gunes, H. and Piccardi, M. (2007). Bi-modal emotion recognition from expressive face and body gestures. *Journal of Network and Computer Applications*, 30(4):1334–1345.

Gurley, K. and Kareem, A. (1999). Applications of wavelet transforms in earthquake, wind and ocean engineering. *Engineering structures*, 21(2):149–167.

Hall, D. and Llinas, J. (1997). An introduction to multisensor data fusion. *Proceedings of the IEEE*, 85(1):6–23.

He, Z., Jin, L., Zhen, L., and Huang, J. (2008). Gesture recognition based on 3d accelerometer for cell phones interaction. In *Circuits and Systems, 2008. APCCAS 2008. IEEE Asia Pacific Conference on*, pages 217–220. IEEE.

Henricksen, K., Indulska, J., and Rakotonirainy, A. (2002). Modeling context information in pervasive computing systems. In *Pervasive Computing*, pages 167–180. Springer.

Henriksen, M., Lund, H., Moe-Nilssen, R., Bliddal, H., and Danneskiod-Samsoe, B. (2004). Test-retest reliability of trunk accelerometric gait analysis. *Gait & posture*, 19(3):288–297.

Hinckley, K., Pierce, J., Sinclair, M., and Horvitz, E. (2000). Sensing techniques for mobile interaction. In *Symposium on User Interface Software and Technology: Proceedings of the 13 th annual ACM symposium on User interface software and technology*, volume 6, pages 91–100. Citeseer.

Horvitz, E., Koch, P., Kadie, C. M., and Jacobs, A. (2002). Coordinate: Probabilistic forecasting of presence and availability. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 224–233. Morgan Kaufmann Publishers Inc.

Huynh, D. T. G. (2008). *Human activity recognition with wearable sensors*. PhD thesis, Technische Universität Darmstadt.

Huynh, T., Blanke, U., and Schiele, B. (2007). Scalable recognition of daily activities with wearable sensors. In *Location-and context-awareness*, pages 50–67. Springer.

Huynh, T., Fritz, M., and Schiele, B. (2008). Discovery of activity patterns using topic models. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 10–19. ACM.

Jenkins, J., Oatley, K., and Stein, N. (1998). *Human emotions: A reader*. Wiley-Blackwell.

Kaila, P. (2008). Oauth and openid 2.0. *From End-to-End to Trust-to-Trust*, 18.

Kapadia, A., Kotz, D., and Triandopoulos, N. (2009). Opportunistic sensing: Security challenges for the new paradigm. In *Communication Systems and Networks and Workshops, 2009. COMSNETS 2009. First International*, pages 1–10. IEEE.

Kapur, A., Kapur, A., Virji-Babul, N., Tzanetakis, G., and Driessen, P. (2005). Gesture-based affective computing on motion capture data. *Affective Computing and Intelligent Interaction*, pages 1–7.

Katz, S., Ford, A. B., Moskowitz, R. W., Jackson, B. A., and Jaffe, M. W. (1963). Studies of illness in the aged: the index of adl: a standardized measure of biological and psychosocial function. *Jama*, 185(12):914–919.

Kazi, S. B., Sikander, S., Yousafzai, S., and Mazhar, S. (2014). Fall detection using single tri-axial accelerometer. In *ASEE 2014 Zone I Conference*.

Khaleghi, B., Khamis, A., Karray, F. O., and Razavi, S. N. (2011). Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*.

Kleres, J. (2011). Emotions and narrative analysis: A methodological approach. *Journal for the theory of social behaviour*, 41(2):182–202.

Ko, M., Cheek, G., Shehab, M., and Sandhu, R. (2010). Social-networks connect services. *Computer*, 43(8):37–43.

Krishnan, N., Juillard, C., Colbry, D., and Panchanathan, S. (2009). Recognition of hand movements using wearable accelerometers. *Journal of Ambient Intelligence and Smart Environments*, 1(2):143–155.

Krishnan, N. C., Colbry, D., Juillard, C., and Panchanathan, S. (2008). Real time human activity recognition using tri-axial accelerometers. In *Sensors, signals and information processing workshop*.

Krumm, J. and Horvitz, E. (2006). Predestination: Inferring destinations from partial trajectories. In *UbiComp 2006: Ubiquitous Computing*, pages 243–260. Springer.

Kwapisz, J. R., Weiss, G. M., and Moore, S. A. (2011). Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82.

Lane, N., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., and Campbell, A. (2010). A survey of mobile phone sensing. *Communications Magazine, IEEE*, 48(9):140–150.

Lara, Ó. D., Pérez, A. J., Labrador, M. A., and Posada, J. D. (2012). Centinela: A human activity recognition system based on acceleration< i> and</i> vital sign data. *Pervasive and mobile computing*, 8(5):717–729.

Lee, J.-M., Kim, Y., Kwon, Y.-S., Derrick, T. R., and Welk, G. J. (2014). Calibration of built-in accelerometer using a commercially available smartphone. In *MEDICINE AND SCIENCE IN SPORTS AND EXERCISE*, volume 46, pages 789–789. LIPPINCOTT WILLIAMS & WILKINS 530 WALNUT ST, PHILADELPHIA, PA 19106-3621 USA.

Lee, K., Cho, Y., and Park, K. (2006). Robust feature extraction for mobile-based speech emotion recognition system. *Intelligent Computing in Signal Processing and Pattern Recognition*, pages 470–477.

Lee, S., Park, H., Hong, S., Lee, K., and Kim, Y. (2003). A study on the activity classification using a triaxial accelerometer. In *Engineering in Medicine and Biology Society, 2003. Proceedings of the 25th Annual International Conference of the IEEE*, volume 3, pages 2941–2943. IEEE.

Lester, J., Choudhury, T., and Borriello, G. (2006). A practical approach to recognizing physical activities. *Pervasive Computing*, pages 1–16.

Liao, L. (2006). *Location-based activity recognition*. PhD thesis, Citeseer.

Lin, C.-H., Wu, N.-Y., Lai, W.-S., and Liou, D.-M. (2014). Comparison of a semi-automatic annotation tool and a natural language processing application for the generation of clinical statement entries. *Journal of the American Medical Informatics Association*, pages amiajnl–2014.

Long, X., Yin, B., and Aarts, R. M. (2009). Single-accelerometer-based daily physical activity classification. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 6107–6110. IEEE.

López, G., Custodio, V., and Moreno, J. (2010). Lobin: E-textile and wireless-sensor-network-based platform for healthcare monitoring in future hospital environments. *Information Technology in Biomedicine, IEEE Transactions on*, 14(6):1446–1458.

Luis, A. and Patricio, M. (2009). Scalable streaming of jpeg 2000 live video using rtp over udp. In *International Symposium on Distributed Computing and Artificial Intelligence 2008 (DCAI 2008)*, pages 574–581. Springer.

Maekawa, T., Kishino, Y., Yanagisawa, Y., and Sakurai, Y. (2012). Mimic sensors: Battery-shaped sensor node for detecting electrical events of handheld devices. In *Pervasive Computing*, pages 20–38. Springer.

Mallat, S. (1999). *A wavelet tour of signal processing*. Access Online via Elsevier.

Markin, M., Harris, C., Bernhardt, M., Austin, J., Bedworth, M., Greenway, P., Johnston, R., Little, A., and Lowe, D. (1997). Technology foresight on data fusion and data processing. *Publication of The Royal Aeronautical Society*.

Mathie, M., Coster, A., Lovell, N., and Celler, B. (2003). Detection of daily physical activities using a triaxial accelerometer. *Medical and Biological Engineering and Computing*, 41(3):296–301.

Maurer, U., Smailagic, A., Siewiorek, D. P., and Deisher, M. (2006). Activity recognition and monitoring using multiple sensors on different body positions. In *Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop on*, pages 4–pp. IEEE.

Mazumder, S. K. (2011). *Wireless networking based control*. Springer.

Michelson, B. (2006). Event-driven architecture overview. *Patricia Seybold Group*.

Miller, G. (2012). The smartphone psychology manifesto. *Perspectives on Psychological Science*, 7(3):221–237.

Miluzzo, E., Lane, N., Fodor, K., Peterson, R., Lu, H., Musolesi, M., Eisenman, S., Zheng, X., and Campbell, A. (2008). Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 337–350. ACM.

MobilizeLabs, U. (2008). Smartphone dataset. [Online; accessed 19-July-2014].

Muñoz, M., Gonzalez, V., Rodríguez, M., and Favela, J. (2003). Supporting context-aware collaboration in a hospital: an ethnographic informed design. *Groupware: Design, Implementation, and Use*, pages 330–344.

Nakatsu, R., Nicholson, J., and Tosa, N. (1999). Emotion recognition and its application to computer agents with spontaneous interactive capabilities. In *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pages 343–351. ACM.

Noble, B., Satyanarayanan, M., Narayanan, D., Tilton, J., Flinn, J., and Walker, K. (1997). Agile application-aware adaptation for mobility. In *ACM SIGOPS Operating Systems Review*, volume 31, pages 276–287. ACM.

Oliver, N., Horvitz, E., and Garg, A. (2002). Layered representations for human activity recognition. In *Multimodal Interfaces, 2002. Proceedings. Fourth IEEE International Conference on*, pages 3–8. IEEE.

Pak, A. and Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of LREC*, volume 2010.

Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.

Patel, S., Mancinelli, C., Bonato, P., Healey, J., and Moy, M. (2009). Using wearable sensors to monitor physical activities of patients with copd: A comparison of classifier performance. In *Wearable and Implantable Body Sensor Networks, 2009. BSN 2009. Sixth International Workshop on*, pages 234–239. IEEE.

Pendão, C. G., Moreira, A. C., and Rodrigues, H. (2014). Energy consumption in personal mobile devices sensing applications. In *Wireless and Mobile Networking Conference (WMNC), 2014 7th IFIP*, pages 1–8. IEEE.

Perich, F. (2002). A service for aggregating and interpreting contextual information. Technical report, Technical report, Hewlett Packard Labs.

Petrovic, S., Osborne, M., and Lavrenko, V. (2010). The edinburgh twitter corpus. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Linguistics in a World of Social Media*, pages 25–26.

Picard, R. (1997). Affective computing.

Picard, R., Vyzas, E., and Healey, J. (2001). Toward machine emotional intelligence: Analysis of affective physiological state. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(10):1175–1191.

Pirttikangas, S., Fujinami, K., and Nakajima, T. (2006). Feature selection and activity recognition from wearable sensors. In *Ubiquitous Computing Systems*, pages 516–527. Springer.

Rahman, M., El Saddik, A., and Gueaieb, W. (2009). Senseface: A sensor network overlay for social networks. In *Instrumentation and Measurement Technology Conference, 2009. I2MTC'09. IEEE*, pages 1031–1036. IEEE.

Rana, R. K., Chou, C. T., Kanhere, S. S., Bulusu, N., and Hu, W. (2010). Ear-phone: an end-to-end participatory urban noise mapping system. In *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 105–116. ACM.

Ravi, N., Dandekar, N., Mysore, P., and Littman, M. (2005). Activity recognition from accelerometer data. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 1541. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Recordon, D. and Reed, D. (2006). OpenID 2.0: a platform for user-centric identity management. In *Proceedings of the second ACM workshop on Digital identity management*, pages 11–16. ACM.

Reilly, J., Ghent, J., and McDonald, J. (2008). *Modelling, classification and synthesis of facial expressions*. Citeseer.

Reiss, A. (2014). *Personalized Mobile Physical Activity Monitoring for Everyday Life*. PhD thesis, Technical University of Kaiserslautern.

Russell, S., Norvig, P., and Intelligence, A. (1995). A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, 25.

Schilit, B. and Theimer, M. (1994). Disseminating active map information to mobile hosts. *Network, IEEE*, 8(5):22–32.

Schlosberg, H. (1954). Three dimensions of emotion. *Psychological review*, 61(2):81.

Schmidt, A., Aidoo, K., Takaluoma, A., Tuomela, U., Van Laerhoven, K., and Van de Velde, W. (1999a). Advanced interaction in context. In *HandHeld and Ubiquitous Computing*, pages 89–101. Springer.

Schmidt, A., Beigl, M., and Gellersen, H. (1999b). There is more to context than location. *Computers & Graphics*, 23(6):893–901.

Schroder, M., Pirker, H., and Lamolle, M. (2006). First suggestions for an emotion annotation and representation language. In *Proceedings of LREC*, volume 6, pages 88–92. Citeseer.

Self-maintenance, P. (1969). Assessment of older people: self-maintaining and instrumental activities of daily living.

Sharkey, J. (2009). Google io 09: Coding for life - battery life, that is. Google.

Shaver, P., Schwartz, J., Kirson, D., and O'connor, C. (1987). Emotion knowledge: further exploration of a prototype approach. *Journal of personality and social psychology*, 52(6):1061.

Shin, M., Cornelius, C., Peebles, D., Kapadia, A., Kotz, D., and Triandopoulos, N. (2011). Anonysense: A system for anonymous opportunistic sensing. *Pervasive and Mobile Computing*, 7(1):16–30.

Shoaib, M., Bosch, S., Incel, O. D., Scholten, H., and Havinga, P. J. (2015). A survey of online activity recognition using mobile phones. *Sensors*, 15(1):2059–2085.

Steinberg, A. N., Bowman, C. L., and White, F. E. (1999). Revisions to the jdl data fusion model. In *AeroSense'99*, pages 430–441. International Society for Optics and Photonics.

Stikic, M., Huynh, T., Van Laerhoven, K., and Schiele, B. (2008). Adl recognition based on the combination of rfid and accelerometer sensing. In *Pervasive Computing Technologies for Healthcare, 2008. PervasiveHealth 2008. Second International Conference on*, pages 258–263. IEEE.

Sumi, Y., Etani, T., Fels, S., Simonet, N., Kobayashi, K., and Mase, K. (1998). C-map: Building a context-aware mobile assistant for exhibition tours. *Community computing and support systems*, pages 137–154.

Tapia, E. M., Intille, S. S., Haskell, W., Larson, K., Wright, J., King, A., and Friedman, R. (2007). Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. In *Wearable Computers, 2007 11th IEEE International Symposium on*, pages 37–40. IEEE.

van Diggelen, F. (2009). *A-GPS: Assisted GPS, GNSS, and SBAS*. Artech House Publishers.

Van Dijk, J. (2012). *The network society*. Sage Publications.

Want, R., Hopper, A., Falcão, V., and Gibbons, J. (1992). The active badge location system. *ACM Transactions on Information Systems (TOIS)*, 10(1):91–102.

Weiser, M. (1991). The computer for the 21st century. *Scientific American*, 265(3):94–104.

Wilson, T., Wiebe, J., and Hoffmann, P. (2009). Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational linguistics*, 35(3):399–433.

Wu, W. H., Bui, A. A., Batalin, M. A., Au, L. K., Binney, J. D., and Kaiser, W. J. (2008a). Medic: Medical embedded device for individualized care. *Artificial Intelligence in Medicine*, 42(2):137–152.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Philip, S. Y., et al. (2008b). Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37.

Yang, J., Wang, J., and Chen, Y. (2008). Using acceleration measurements for activity recognition: An effective learning algorithm for constructing neural classifiers. *Pattern recognition letters*, 29(16):2213–2220.

Zimmermann, A., Lorenz, A., and Oppermann, R. (2007). An operational definition of context. In *Modeling and using context*, pages 558–571. Springer.